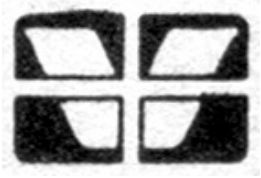
HISOFT DEVPAC

(ZX-SPECTRUM 48K / PLUS)

Ensamblador / Desensamblador
Editor Código Máquina Z80



Producido y distribuido
en exclusiva en España por:

VENTAMATIC
c/ Córcega, 89, entlo.
08029 BARCELONA

Es imprescindible leer íntegramente el contenido de este manual antes de empezar a utilizar el programa.

VEA EL APARTADO SOBRE CONSULTAS EN LA CONTRAPORTADA.

HISOFT DEVPAC

Ensamblador-desensamblador-editor de código máquina (ZX-SPECTRUM 48K). Copyright: HISOFT. Producido y distribuido en exclusiva en España por VENTAMATIC - c/ Córcega, 89, entlo. - 08029 BARCELONA

PUESTA A PUNTO.

GENS3 es un poderoso y manejable ensamblador del Z80, muy cercano en definición al ensamblador Zilog. De forma distinta a otros muchos ensambladores disponibles para microordenadores, GENS3 es un extenso programa profesional, y le animamos para que estudie las siguientes secciones junto con el ejemplo que se incluye, muy cuidadosamente antes de intentar trabajar con el ensamblador. Si es completamente nuevo en el uso del código máquina, trabaje primero con el ejemplo.

GENS3 tiene una longitud de 10043 bytes, una vez relocalizado, y utiliza su propia pila interna. Se suministra un editor integral de líneas el cual sitúa el fichero de texto inmediatamente después del código del ensamblador, mientras la tabla de símbolos se crea después del fichero de texto: Así, cuando cargue GENS3, debe dejar suficiente espacio para incluir el mismo ensamblador y el máximo tamaño para la tabla de símbolos y el texto que quiera utilizar en la sesión en curso. Será conveniente, por lo tanto, cargar GENS3 en la parte baja de memoria.

Para cargar GENS3 proceda como sigue:

Coloque la cinta suministrada en su reproductor de cassette, escriba **LOAD "" CODE xxxxx** y pulse PLAY en el reproductor - donde xxxxx es la dirección decimal en la cual quiere que GENS3 se ejecute.

Una vez ha cargado el código de GENS3 en el ordenador, puede entrar en el ensamblador con **RANDOMIZE USR xxxxx** <ENTER> donde xxxxx es la dirección en la cual cargó el ensamblador. Si posteriormente desea reentrar en el ensamblador, ejecute **RANDOMIZE USR *dirección + 56*** para una "entrada en frío" (destruyendo el texto) o **RANDOMIZE USR *dirección + 61*** para una "entrada en caliente" (conservando el texto anteriormente creado).

Por ejemplo, digamos que quiere cargar GENS3 para que se ejecute en la dirección 26000 - proceda como sigue:

LOAD "" CODE 26000
RANDOMIZE USR 26000

Para reentrar en el ensamblador utilice **RANDOMIZE USR 26056** para una "entrada en frío" o **RANDOMIZE USR 26061** para una "entrada en caliente".

El control pasará ahora al editor: se producirá un mensaje de confirmación y aparecerá la señal de editor '**>**'.

Antes,sin embargo, lea todo este manual para conocer todas las posibilidades que le ofrece GENS3 a fondo.

VARIACIONES EN LA ULTIMA VERSION DE GENS3

El Comando 'C'

El comando del editor 'C' no realiza una compactación del fichero de texto pero en su lugar le permite *'**C**'onfigurar* los tamaños de los dos buffers utilizados por el ensamblador.

El primer buffer es el buffer Include en el cual se almacena el texto cuando ensamblamos desde la cinta de cassette o el ZX Microdrive - cuanto más largo sea este buffer, más texto se leerá desde la cinta o el Microdrive cuando utilicemos la opción de inclusión ***F**. Un gran buffer significará normalmente un ensamblado rápido pero la desventaja es que se consumirá más memoria por cada bloque de texto. Así, hay que comprometerse cuando componemos, especialmente desde Microdrive, entre la rapidez de ensamblado y la utilización de memoria, y el comando '**C**' le da la capacidad de controlar este convenio asignando el tamaño del buffer Include en bytes, con un mínimo de 256 bytes. El segundo buffer para el cual debe especificar el tamaño es el buffer Macro. Este se usa para almacenar el texto de cualquier definición de macro, y así podrá hacerlo lo suficientemente grande para almacenar el texto de todos los macros que va a declarar en esta sesión.

El comando 'C' le invita a entrar el tamaño del *'buffer Include'* y después el tamaño del *'buffer Macro'*. En ambos casos simplemente entre el número de bytes (en decimal) que desea asignar, seguido por <ENTER>. Si sólo pulsa <ENTER> no se producirá ninguna acción. Si especifica un tamaño de buffer Include entonces dicho tamaño forzosamente ha de ser de un mínimo de 256 bytes. Puede anular este comando pulsando CAPS SHIFT 1.

Notará que la utilización del comando '**C**' efectivamente, comporta una "entrada en frío" de su texto, borrándolo de la memoria. Así, acuérdese de salvar cualquier texto existente antes de utilizar el comando '**C**'.

La función previa del comando '**C**' (la función de 'C'ompactación) ha sido reemplazada por el nuevo comando '**Q**'.

El Comando 'H'

Este comando le permite verificar un fichero de texto que ha sido salvado en un cartucho de Microdrive. Utilice simplemente **H, ,n:nombre-de-fichero** donde **n** es el número de drive en el que has salvado el fichero y **nombre-de-fichero** es el nombre del fichero requerido del drive. El fichero se abre y cada registro se inspecciona para ver si el registro se puede leer desde el drive. Los números de registro se muestran al tiempo que son comprobados. Si un número de registro en particular no está presente se mostrará *'File not found'* y el control volverá al BASIC. Si el fichero se verifica, el control vuelve al editor del ensamblador.

Macros

Un macro toma la siguiente forma:

```
Nombre   MAC
         .
         .
         definición de macro
         .
         .
         ENDM
```

Donde **Nombre** es el nombre del macro que invocará al texto del macro cada vez que se utilice Nombre en el campo de nemónicos, **MAC** define el principio del la definición de macro y **ENDM** indica el final de la definición. Los parámetros (hasta 32) pueden referenciarse en la definición de macro con la utilización de signos de igual '=' seguido por el número de parámetro (0-31 inclusive).

Por ejemplo el macro:

```
MOVE   MAC
       LD HL,=0
       LD DE,=1
       LD BC,=2
       LDIR
       ENDM
```

toma tres parámetros,dirección fuente, dirección de destino y longitud, carga los valores pertinentes en HL, DE y BC y entonces ejecuta la instrucción LDIR. Para llamar al macro posteriormente utilice simplemente el nombre del macro en el campo de nemónicos seguido por los valores que deseas que tomen los tres parámetros p.e.

```
         MOVE 16384,16385,4096
```

Hemos utilizado direcciones especificas en este ejemplo, pero de hecho, podemos usar cualquier expresión válida para especificar el valor de un parámetro de macro p.e.

```
         MOVE start,start+1,long
```

Dentro de la definición de macro, los parámetros pueden aparecer en cualquier expresión válida, por ejemplo:

```
HMS    MAC

       LD HL,=0*3600
       LD DE,=1*60
       ADD HL,DE
       LD DE,=2
       ADD HL,DE

       ENDM
```

es una macro que toma tres parámetros - horas,minutos,segundos, y produce en el registro HL el número total de segundos especificados por los parámetros. Podría usarla así:

```
Horas      EQU 2
Minutos    EQU 30
Segundos   EQU 12
Inicio     EQU 0

       HMS Horas,Minutos,Segundos
       LD DE,Inicio
       ADD HL,DE    ;HL da el tiempo final
```

Observe que los macros no pueden anidarse, por lo tanto no puede definir un macro dentro de una definición de macro ni puede invocar a una macro desde una definición de macro.

Durante el ensamblado, siempre que se encuentra un nombre de macro en el campo de nemónicos, el texto del macro se ensambla. Normalmente este texto no se mostrará en el listado de lo ensamblado - solo se muestra el nombre de macro. Sin embargo, puede forzar el listado de cualquier macro usando el comando del ensamblador ***M+** antes de lo que quiera listar - utilice posteriormente ***M-** si quiere que quede de nuevo desactivado el listado de macros.

Si sobrepasa el espacio de Buffer Macro, se mostrará un mensaje y el ensamblado se detendrá. Use el comando '**C**' después de salvar su texto, para asignar un buffer Macro mayor.

Ahora se generan algunos errores extra de ensamblado. Estos son:

```
*Error*16 Nested macro definition
*Error*17 This identifier is not a macro
*Error*18 Nested macro call
*Error*19 Nested conditional statement
```

DETALLES DEL GENS3.

Cómo trabaja GENS3.

GENS3 es un rápido ensamblador Z80 de dos pasos él cual es fácilmente adaptable para su ejecución en la mayoría de los sistemas Z80. El ensamblador ensambla todos los nemónicos standard del Z80 y añade ventajas extra que incluyen el ensamblado condicional, muchos comandos del ensamblador y una tabla dé símbolos binaria.

Cuando ensamble (usando el comando '**A**' del editor - ver el Editor) se le preguntará primero el *'Table size:'* (Tamaño de la tabla) en decimal. Esto es la cantidad de espacio que se asignará a la tabla de símbolos durante el ensamblado. Si pulsa simplemente <ENTER>, GENS3 elegirá un tamaño de tabla que crea adecuado para el tamaño del texto - normalmente será suficiente. Observe qué cuando utilizamos la opción *'Include'* puede que tenga que especificar un tamaño para la tabla de símbolos mayor de lo normal; el ensamblador no puede predecir el tamaño del fichero que será incluido.

Despues de *'Table size:'* se le preguntará por algunas *'Options:'* (Opciones) que pueda necesitar. Entre éstas en decimal sumando los números dé opción juntos si quiere más de una opción. Las opciones disponibles son:

Opción 1	produce un listado de la tabla de símbolos al final de la segunda pasada del ensamblado.
Opción 2	No genera ningún código objeto.
Opción 4	No produce un listado del ensamblado.
Opción 8	Dirige el listado del ensamblado a la impresora.
Opción 16	Simplemente sitúa el código objeto, si se genera, después de la tabla de símbolos. El contador de posición está todavía señalada por ORG, por tanto él código objeto puede situarse en una sección dé memoria pero designarse para su ejecución en cualquier otra.
Opción 32	Desactiva el chequeo de donde va el código objeto- útil para acelerar el ensamblado.

Ejemplo: La *Opción 36* produce un ensamblado rápido - no se genera ningún listado ni se realizan comprobaciones para ver donde se sitúa el código objeto.

Observe que si ha usado la *Opción 16*, el pseudo-nemónico del ensamblador ENT no tendrá efecto. Puede averiguar donde se ha situado el código objeto, si ha utilizado la *Opción 16*, usando él comando del editor '**X**' para encontrar el final del texto (el segundo número mostrado) y después añadiéndole el espacio asignado a la tabla de símbolos + 2.

El ensamblado tiene lugar en dos pasadas; durante la primera pasada GENS3 busca los errores y compila la tabla de símbolos,

la segunda pasada genera código objeto (si no se especifica la O*pción 2*). Durante el primer paso no se muestra nada en la pantalla ni en la impresora a menos que sé detecte un error, en cuyo caso la línea implicada se imprimirá con un número de error siguiéndole (ver Apéndice 1). El ensamblado se detiene - pulse '**E**' para volver al editor o cualquier otra tecla para continuar el ensamblado desde la siguiente línea.

Al final del primer paso el mensaje:

Pass 1 errors: no

se imprimirá. Si se han detectado algunos errores, se detendrá el ensamblado y no se realizará el segundo paso. Si se hacía referencia a alguna etiqueta en el campo dé operandos pero no está declarada en un campo de etiquetas, se mostrará el mensaje *'*WARNING*' label absent'* para cada olvido de declaración de etiqueta.

Es durante el segundo paso cuando se genera el código objeto (a no ser que la generación haya sido desactivada por la *Opción 2* - ver anteriormente). Se genera un listado del ensamblador durante este paso, a menos que haya sido desactivado por la *Opción 4* o por el comando ***L-** del ensamblador. El listado del ensamblador es normalmente de la forma:

```
0000 210100      25 label
                    LD   HL,1
1    6        15    21   26
```

La primera anotación de la línea es el valor del Contador de Posición al principio del procesamiento de esta línea, a menos que el nemónico de esta línea sea uno de los pseudo-nemónicos ORG, EQU o ENT (ver Sección 2.6) en cuyo caso la primera anotación representará el valor en el campo de operandos de la instrucción. Esta anotación sé muestra normalmente en hexadecimal pero puede mostrarse en decimal sin signo utilizando el comando ***D+** del ensamblador (ver comandos del ensamblador).

La siguiente anotación, desde la columna 6, es de un máximo de 8 caracteres de longitud (representando un máximo de 4 bytes) y es el código objeto producido por la instrucción en curso - (ver el comando ***C** del ensamblador más adelante).

Después viene el número de línea - un entero en el rango de 1 a 32767 inclusive.

Las columnas 21-26 de la primera línea contienen los primeros 6 caracteres de cualquier etiqueta definida en ésta línea.

Despues de cualquier etiqueta viene una nueva línea - en esta

línea el nemónico se muestra desde la columna 21 a 24. Después viene el campo de operandos desde la columna 26 de esta línea y finalmente cualquier comentario que haya sido insertado al final de la línea con generación de nuevas líneas si es necesario.

El formato anterior facilita la lectura del listado del ensamblador en la pantalla del SPECTRUM. GENS3 no redefine la anchura de pantalla del SPECTRUM porque esto incrementaría el espacio ocupado por él y sería restrictivo ya que no podrían utilizarse las rutinas standard de salida de la ROM del SPECTRUM.

El comando ***C** del ensamblador puede utilizarse para producir un listado reducido del ensamblado - su efecto es omitir los 9 caracteres que representan el código objeto de la línea, permitiendo así que se alojen más líneas de ensamblador en una línea de pantalla. Ver comandos del ensamblador más adelante.

Es posible modificar la forma en la cual cada línea del listado es dividida con el POKE a 3 posiciones de GENS3. Detalles de como hacer esto se dan más adelante. Distinguimos entre *'línea de ensamblado'* la cual es la línea en curso del listado del ensamblado almacenada en un buffer interno, y *'línea de pantalla'* que es la línea que aparece actualmente en la pantalla. Una línea de ensamblado generará normalmente más de una línea de pantalla.

1. La posición *'Principio de GENS3 + 51 (#33)'*, almacena en qué posición de columna - 5 terminará la primera línea de pantalla de la línea de ensamblado. Cambie este byte a cero para provocar que la línea sea desechada o cualquier otro valor (<256) para que la línea de pantalla acabe en una columna en particular.

2. La posición *'Principio de GENS3 + 52 (#34)'*, almacena la columna (empezando en 1) en la que empezará cada línea de pantalla de la línea de ensamblado.

3. La posición *'Principio de GENS3 + 53 (#35)'*, almacena cuántos caracteres del resto de la línea de ensamblado se van a mostrar en cada línea de pantalla.

Como un ejemplo, digamos que quiere que la primera línea de pantalla de cada línea de ensamblaje contenga 20 caracteres (es decir, no incluyendo el campo de etiquetas) y después que cada línea de pantalla siguiente empiece en la columna 1 y llene la línea completa. También asumimos que has cargado GENS3 en #5E00 ó 24064 decimal. Para efectuar estos cambios, ejecute las siguientes instrucciones POKE desde BASIC:

```
POKE 24115,20
POKE 24116,1    debe haber como mínimo un espacio al principio
POKE 24117,31   de cada subsiguiente línea de pantalla
```

Las modificaciones anteriores son solo aplicables si el comando ***C** no se ha utilizado - la utilización del comando ***C** provoca que las líneas sean colocadas donde sea necesario.

El listado del ensamblado puede ser detenido al final de una línea pulsando **CAPS SHIFT** y **SPACE** juntos - posteriormente pulse '**E**' para volver al editor o alguna otra tecla para continuar el listado.

Los únicos errores que se pueden producir durante el segundo paso son '**ERROR* 10*' (ver Apéndice 1) y '*Bad ORG!*' (que se produce cuando el código objeto intenta sobreescribir GENS3, el fichero de texto o la tabla de símbolos - la detección de éste puede desactivarse con la *Opción 32*). '**ERROR* 10*' no es fatal y puedes continuar el ensamblado como con los errores del primer paso, mientras que '*Bad ORG!*' es fatal e inmediatamente devuelve el control al editor.

Al final del segundo paso se mostrará el mensaje:

Pass 2 errors: no

seguido de avisos de alguna etiqueta ausente. Luego se muestra el siguiente mensaje:

Table used:xxxxx from yyyyy

Esto informa sobre la cantidad de tabla de símbolos utilizada comparada con la cantidad que se asignó.

En este punto, si el pseudo-nemónico ensamblador ENT se ha usado correctamente, se mostrará el mensaje '*Executes:nnnnn*'. Esto muestra la dirección de ejecución del código objeto - puede ejecutar el código usando el comando '**R**' del editor. Tenga cuidado de utilizar el comando '**R**' a menos que haya acabado un ensamblado correctamente y haya visto el mensaje '*Executes:nnnnn*'.

Finalmente, si se ha especificado la *Opción 1*, se producirá un listado alfabético de las etiquetas usadas y sus valores asociados. El número de anotaciones mostradas en una línea se puede cambiar con un **POKE inicio de GENS3 + 50, con el valor pertinente**; el valor por defecto es 2.

El control vuelve ahora al editor.

Formato de Instrucción en Ensamblador.

Cada línea de texto a procesar por GENS3 debe tener el siguiente formato donde algunos campos son opcionales:

ETIQUETA	**NEMONICO**	**OPERANDOS**	**COMENTARIO**
Start	LD	HL, label	;coge 'label'

Los espacios y caracteres de tabulación (insertados por el editor) generalmente se ignoran.

La línea se procesa de la siguiente forma:

Se revisa el primer caracter de la línea, y las siguientes acciones dependen de la naturaleza de ese caracter como se indica abajo:

';' la línea completa es tratada como un comentario, es decir ignorada.

'*' espera el siguiente(s) caracter(es) para constituir un comando ensamblador (ver comandos del ensamblador). Trata todos los caracteres siguientes al comando como un comentario.

' ' (caracter de fin de línea) simplemente ignora la línea.

'' (espacio o tab) si el primer caracteres un espacio o un carácter de tabulación, GENS3 espera al siguiente carácter no-espacio o no-tab como principio de un nemónico del Z80.

Si el primer caracter de una línea es algún caracter distinto de los dados anteriormente, el ensamblador espera una etiqueta (ver Etiquetas).

Después de procesar una etiqueta válida, o si el primer caracter de una línea es un *espacio/tab*, el ensamblador busca el siguiente caracter *no-espacio/tab* y espera que esto sea un caracter de fin de línea o un nemónico del Z80 (ver Apéndice 2) de un máximo de 4 caracteres de longitud y terminado con un caracter de *espacio/tab* o de fin de línea. Si el nemónico es válido y requiere uno o más operandos, los *espacios/tabs* se saltan y se procesa el campo de operandos.

Las etiquetas pueden estar presentes solas en una instrucción ensamblador; muy útil para aumentar la legibilidad del listado. Los comentarios pueden aparecer en cualquier lugar después del campo de operandos o, si un nemónico no tiene argumentos, después del campo de nemónicos.

Etiquetas.

Una etiqueta es un símbolo que representa hasta 16 bits de información.

Una etiqueta puede usarse para especificar la dirección de una instrucción particular o un área de datos o puede utilizarse como una constante por medio del pseudo-nemónico EQU (ver pseudo-nemónicos del ensamblador).

Si una etiqueta está asociada con un valor mayor de 8 bits y se usa en un contexto donde es aplicable una constante de 8 bits, el ensamblador generará un mensaje de error p.e.

```
label EQU 1234
      LD A,label
```

generará un '**ERROR*10*' cuando se procese la segunda instrucción durante el segundo paso de ensamblado.

Una etiqueta puede contener cualquier número de caracteres válidos, aunque solo los 6 primeros serán tratados como significativos; estos primeros 6 caracteres deben ser únicos ya que una etiqueta no puede redefinirse (**ERROR*4*). Una etiqueta no debe constituir una palabra reservada (ver Apéndice 2) aunque una palabra reservada puede ser tomada como parte de una etiqueta.

Los caracteres que se pueden utilizar en una etiqueta son 0-9, $ y A-z. Observa que 'A-z' incluye todos las mayúsculas y minúsculas junto con los caracteres [, \,], ^, £ y _. Una etiqueta debe empezar con un caracter alfabético. Algunos ejemplos de etiquetas válidas son:

```
LOOP
loop
a_long_label
L 1
L 2
a
LDIR    LDIR no es una palabra reservada.
two 5
```

Contador de posición.

El ensamblador mantiene un Contador de Posición, así un símbolo del campo de etiquetas puede asociarse con una dirección y entrarse en la Tabla de Símbolos. Este Contador de Posición puede ser asignado a cualquier valor por medio del pseudo-nemónico ORG.

El símbolo '**$**' se puede utilizar para hacer referencia al valor actual del Contador de Posición p.e. *LD HL,$+5* genera código que cargaría el par de registros HL con un valor de 5 más el valor actual del Contador de Posición.

Tabla de Símbolos.

Cuando se encuentra una etiqueta por primera vez, se entra en la tabla con dos punteros que indican en un momento posterior, como se relaciona esta etiqueta alfabéticamente con otras etiquetas de la tabla. Si la primera aparición de la etiqueta es en el campo de etiquetas, su valor (dado por el Contador de Posición o por el valor de la expresión siguiente a un pseudo-nemónico EQU del ensamblador) se entra en la Tabla de Símbolos. En otro caso, el valor se entra siempre que se encuentre el símbolo en el campo de etiquetas.

Este tipo de tabla de símbolos se llama Tabla de Símbolos binaria y su estructura permite entrar símbolos y recuperarlos de la tabla en un muy corto espacio de tiempo (esencial para grandes programas). El tamaño de una anotación en la tabla varía desde 8 bytes a 13 bytes dependiendo de la longitud del símbolo.

Si durante la primera pasada de ensamblado, se define un símbolo más de una vez, se genera un (**ERROR* 4*) ya que el ensamblador no sabe que valor debería asociar a dicho símbolo.

Si un símbolo no es asociado con un valor, se generará el mensaje '**WARNING* symbol absent*' al final del ensamblado. La ausencia de una definición de símbolo no evita que continúe el ensamblado.

Observe que solo se entran en la Tabla de Símbolos los 6 primeros caracteres de un símbolo, en orden a reducir el tamaño de la tabla.

Al final del ensamblado recibirá un mensaje informando sobre cuanta memoria fue utilizada por la Tabla de Símbolos durante este ensamblado (puede cambiar la cantidad de memoria asignada a la Tabla de Símbolos respondiendo a la pregunta '*Table:*' cuando empieza el ensamblado).

Expresiones.

Una expresión es un operando consistente en bien un término simple o bien una combinación de términos separados entre sí por un operador. Las definiciones de término y operador son:

TERMINO	constante decimal p.e. 1029 constante hexadecimal p.e. #405 constante binaria p.e. %10000000101 constante tipo caracter p.e. "a" etiqueta p.e. L1029 también se puede utilizar '$' para referirse al valor actual del Contador de Posición.
OPERADOR	'+' suma '-' resta '&' AND lógico '@' OR lógico '!' XOR lógico '*' multiplicación entera '/' división entera '?' función MOD (a?b= a - (a/b)*b)

Notas: '#' se usa para indicar el principio de un número hexadecimal, '%' para un número binario y '"' para una constante caracter. Cuando lee un número (decimal, hexadecimal o binario) GENS3 toma los 16 bits menos significativos del número (es decir MOD 65536) ej.: 70016 se convierte en 4480 y #5A2C4 se convierte en #A2C4.

Se han proporcionado una amplia variedad de operandos pero no se observa ningún operador de prioridad - las expresiones son evaluadas estrictamente de izquierda a derecha. Los operadores '*', '/' y '?' se proporcionan simplemente como conveniencia y no para formar parte de un manipulador de expresiones complejas, lo cual incrementaría el tamaño de GENS3.

Si una expresión está encerrada entre paréntesis, se toma como la representación de direcciones de memoria como en la instrucción *LD HL,(loc+5)* que cargaría el par de registros HL con el valor de 16 bits contenido en la posición de memoria '*loc+5*'.

Algunas instrucciones del Z80 (JR y DJNZ) esperan operandos que tengan un valor de 8 bits y no de 16 bits esto se llama direccionamiento relativo. Cuando especifican direcciones relativas, GENS3 automáticamente resta el valor del Contador de Dirección en la siguiente instrucción desde el valor dado en el campo de operandos de la instrucción en curso, para obtener la dirección relativa de la instrucción actual.

El rango de valores permitidos como direcciones relativas son el valor del Contador de Posición de la siguiente instrucción -128 a +127.

Si, en su lugar, desea especificar un salto relativo desde el valor del Contador de Posición de la instrucción actual, utilice el símbolo '**$**' (palabra reservada) seguida por el desplazamiento requerido. Debido a que ahora es relativo a la instrucción actual del Contador de Posición, el desplazamiento debe estar en el rango -126 a +129 inclusive.

Ejemplos de expresiones válidas

```
#5000 - etiqueta
%1001101 ! %1011        da %1000110
#3456 ? #1000           da #456
4 + 5 * 3 - 8           da 19
$ - etiqueta + 8
2345 / 7 - 1            da 334
"A"+128
"y"-";"+7
(5 * label - #1000 & %1111)
17 @ %1000              da 25
```

Observe que los espacios pueden insertarse entre términos y operadores y viceversa pero no entre términos.

Si una operación de multiplicación produjese un valor absoluto mayor de 32767 se mostraría '*ERROR* 15' mientras si una operación de división implica una división por cero se dará un '*ERROR* 14' - en otro caso el overflow se ignora. Todas las utilizaciones aritméticas del complemento a dos, donde cualquier número mayor de 32767 es tratado como negativo p.e. 60000 = -5536 (60000 - 65536).

Pseudo-nemónicos del ensamblador.

Ciertos 'pseudo-nemónicos' son reconocidos por GENS3. Estos pseudo-nemónicos del ensamblador, como se les llama, no tienen efecto sobre el procesador Z80 en el momento de la ejecución, es decir, no son decodificados en códigos de operación, simplemente dirigen al ensamblador a realizar ciertas acciones en el momento de ensamblar. Estas acciones tienen el efecto de cambiar, de alguna forma, el código objeto producido por GENS3.

Los pseudo-nemónicos son ensamblados exactamente como instrucciones ejecutables; pueden ir precedidos por una etiqueta (necesario para EQU) y seguidos por un comentario. Los pseudo-nemónicos disponibles son:

ORG expresión

Asigna el valor de '*expresión*' al Contador de Posición. Si la *Opción 2* y la *Opción 16* no estan seleccionadas y un *ORG* produjese la sobreescritura del programa GENS, el fichero de texto o la tabla de símbolos, se muestra el mensaje *'Bad ORG*' y se detiene el ensamblado. Ver "Como trabaja GENS3" para más detalles de como las Opciones 2 y 16 afectan el uso de ORG.

EQU expresión

Debe ir precedido por una etiqueta. Asigna el valor de '*expresión*' al valor de la etiqueta. La expresión no puede contener un símbolo al que no haya sido ya asignado un valor (*ERROR*13*).

DEFB expresión,expresión.....

Cada '*expresión*' debe evaluar 8 bits; el byte de la dirección apuntada por el Contador de Posición se activa al valor de '*expresión*' y el Contador se incrementa en 1. Repite para cada expresión.

DEFW expresión,expresión.....

Asigna la '*palabra*' (dos bytes)de la dirección actualmente apuntada por el Contador de Posición al valor de '*expresión*' y avanza el Contador de Posición en 2. El byte menos significativo se sitúa primero seguido por el byte más significativo. Repite para cada expresión.

DEFS expresión

Incrementa el Contador de Posición por el valor de '*expresión*' - equivalente a reservar un bloque de memoria de tamaño igual al valor de '*expresión*'.

DEFM "s"

Define el contenido de '**n**' bytes de memoria para que sean iguales a la representación ASCII de la cadena '**s**', donde '**n**' es la longitud de la cadena y puede ser en teoría, del rango de 1 a 255 inclusive, aunque en la práctica, la longitud de la cadena está limitada por la longitud de la línea que puede entrar desde el editor. El primer caracter del campo de operandos ('"' anterior) es cogido como el delimitador de la cadena y la cadena **s** se define como los caracteres encerrados entre dos delimitadores; el caracter de fin de línea también actúa como final de la cadena.

ENT expresión

asigna a la dirección de ejecución del código objeto ensamblado el valor de la expresión - utilizado en conjunción con el comando '**R**' del editor. No hay valor por defecto para la dirección de ejecución.

Pseudo-nemónicos condicionales.

Los pseudo-nemónicos condicionales proporcionan al programador la capacidad de incluir o no incluir ciertas secciones del texto fuente en el proceso de ensamblado. Esto se hace posible a través del uso de IF, ELSE y END.

IF expresión

evalúa '*expresión*'. Si el resultado es cero, el ensamblado de las siguientes líneas se desactiva hasta que se encuentre bien un pseudo-nemónico '*ELSE*' o bien un '*END*'. Si el valor de '*expresión*' es distinto de cero, el ensamblado continúa normalmente.

ELSE

este pseudo-nemónico simplemente activa y desactiva el ensamblado. Si el ensamblado estaba activado antes de encontrar el '*ELSE*', será posteriormente desactivado y viceversa.

END

'*END*' simplemente activa el ensamblado.

Nota: Los pseudo-nemónicos condicionales no pueden anidarse; no se realiza ninguna comprobación para **IFs** anidados, por tanto cualquier intento de anídamiento de estos nemónicos obtendrá resultados inesperados.

Comandos del ensamblador.

Los comandos del ensamblador, como los pseudo-nemónicos, no tienen efecto en el procesador Z80 en el momento de la ejecución ya que no están decodificados en códigos de operación. Sin embargo, de forma distinta a los pseudo-nemónicos del ensamblador, tampoco tienen efecto sobre el código objeto producido por el ensamblador - los comandos del ensamblador simplemente modifican el formato del listado.

Un comando del ensamblador es una línea del texto fuente que empieza con un asterisco '*****' . La letra siguiente al asterisco determina el tipo de comando y debe estar en modo mayúsculas. El resto de la línea puede ser cualquier texto excepto los comandos '**L**' y '**D**' que esperan un '**+**' o un '**-**' después del comando.

Están disponibles los siguientes comandos:

***E**

(*eject*) produce que se manden tres líneas en blanco a la pantalla o a la impresora - útil para separar módulos.

***Hs**

Produce que la cadena '**s**' sea tomada como encabezamiento a imprimir después de cada '*eject*' (***E**). ***H** automáticamente ejecuta un ***E**.

***S**

Provoca que el listado se detenga en esta línea. El listado puede reactivarse pulsando alguna tecla. Es útil para leer las direcciones en medio del listado. Nota: Aunque ***S** es siempre reconocido, despues de un ***L-**, ***S** no detendrá la impresión.

***L-**

Provoca que el listado y la impresión sean desconectados empezando con esta línea.

***L+**

Provoca que el listado y la impresión sean activados empezando con esta línea.

***D+**

Provoca que el valor del Contador de Posición sea dado en decimal al principio de cada línea en lugar del normal hexadecimal. Se utilizará decimal sin signo.

***D-**

Vuelve a la utilización del hexadecimal para el valor del Contador de posiciones al principio de cada línea.

***C-**

Acorta el listado del ensamblador empezando desde la siguiente línea. El listado se abrevia no incluyendo el display del código objeto generado por la línea en curso. Esto salva 9 caracteres y permite escribir más líneas de ensamblador, mejorando así la legibilidad.

***C+**

Vuelve al listado completo del ensamblador como se describe en la Sección 'Cómo trabaja GENS3".

***F nombre-fichero**

Este es un comando muy potente que le permite ensamblar texto desde cinta (Vea el apartado USO DEI MICRODRIVE). El fichero de texto es leido desde la cinta en un buffer, un bloque cada vez, y después ensamblando desde el buffer; esto le permite crear grandes cantidades de código objeto ya que el texto que está siendo ensamblado, no ocupa el valioso espacio de memoria.

El nombre (máximo 10 caracteres) del fichero de texto que desea 'incluir' en este punto en el ensamblado puede, opcionalmente, ser especificado después de la '**F**' y debe ir precedido por un espacio. Si no se da ningún nombre de fichero, se incluirá el primero que se encuentre en la cinta.

Cualquier fichero de texto que desee incluir mediante esta opción debe haber sido previamente volcado en cinta usando el comando '**T**' del editor y no el comando '**P**' - esto es necesario porque un fichero de texto a incluir debe ser volcado en bloques con suficiente longitud de espacios inter-bloques para permitir el ensamblado del bloque actual antes de que se cargue el siguiente desde la cinta. El tamaño de este bloque es determinado por el tamaño del buffer *Include* el cual puede asignarse mediante el comando '**C**' (Le remitimos a las notas siguientes a la Puesta a punto).

La capacidad de seleccionar el tamaño de este buffer (tanto para Microdrive como para cassette) le permite optimizar la proporción tamaño/velocidad de cualquier inclusión de texto desde cinta o Microdrive; por ejemplo, si no se propone usar el comando '**F**' durante una sesión de ensamblado, puede encontrar útil especificar un tamaño muy pequeño de buffer para minimizar el espacio tomado por GENS3 y su área de trabajo.

Observe que el tamaño de buffer especificado en la sesión en la cual volcó el fichero a incluir ha ser el mismo que el tamaño del buffer dado en la sesión en la que está actualmente incluyendo el texto.

Siempre que el ensamblador detecta un comando '**F**' sugiere '*Start tape..*', esto ocurrirá en la primera y segunda pasada ya que el texto incluido debe ser explorado en cada paso. La cinta es entonces 'rastreada' para encontrar un fichero '*Include*' con el nombre requerido, o el primer fichero. Si se encuentra un fichero cuyo nombre no coincide con el requerido, se mostrará el mensaje '*Found nombre-de-fichero*' y la búsqueda continuará, en otro caso se mostrará '*Using nombre-de-fichero*' , el fichero se cargará, bloque por bloque, y se incluirá.

Ver Apéndice 3 para un ejemplo de la utilización de este comando.

Los comandos del ensamblador, distintos de ***F**, son reconocidos solo en la segunda pasada.

Si el ensamblado se ha desactivado por un pseudo-nemónico condicional, el efecto de cualquier comando del ensamblador también se desactiva.

EL EDITOR INTEGRAL,

Introducción al Editor.

El editor suministrado con todas las versiones de GENS3 es un simple editor basado en líneas diseñado para trabajar con todos los sistemas operativos Z80 al mismo tiempo que para el mantenimiento fácil de uso y la capacidad de editar programas rápida y eficientemente.

En orden a reducir el tamaño del fichero de texto, el editor realiza una cierta compresión de espacios. Esto tiene lugar de acuerdo con el siguiente modelo: siempre que se escribe una línea desde el teclado se entra, caracter a caracter, en un buffer interno al ensamblador; entonces, cuando se termina la línea (es decir, cuando pulsa **ENTER**), se transfiere desde el buffer al fichero de texto. Es durante esta transferencia que ciertos espacios se comprimen: la línea se examina desde su primer caracter, si es un espacio, se entra un caracter *TAB* en el fichero de texto y todos los espacios siguientes se saltan. Si el primer caracter no es un espacio, los caracteres se transfieren desde el buffer al fichero hasta que se detecta un espacio, donde la acción a tomar será la misma que si el siguiente caracter fuera el primer caracter de la línea. Esto se repite posteriormente con el resultado de que se insertan caracteres *TAB* al principio de la linea o entre la etiqueta y el nemónico y entre el nemónico y los operandos y entre los operandos y los comentarios. Por supuesto, si se detecta en cualquier momento algún caracter de retorno de carro (**ENTER**), la transferencia acaba y el control vuelve al editar.

Este proceso de compresión es transparente al usuario, que puede usar simplemente caracteres de control de tabulación (CI - ver más adelante) para producir un fichero de texto limpiamente tabulado, el cual será al mismo tiempo económico en almacenamiento.

Observe que los espacios no se comprimen en los comentarios y no estarán presentes en una etiqueta, nemónico o campo de operandos.

A través de esta sección se utilizan ciertas abreviaciones para referirse a códigos de control particulares que son:

ENTER	ENTER del SPECTRUM.
CC	CAPS SHIPT y 1. Utilizado para anular una entrada.
CH	DELETE o CAPS SHIFT y 0. Borrado.
CI	CAPS SHIFT y 8. Avanza a la siguiente posición TAB.
CX	CAPS SHIFT y 5. Olvida la última línea entrada.

Se entra en el editor automáticamente cuando se ejecuta GENS3, y se muestra el mensaje:

seguido del indicador de editor '>'.

En respuesta al indicador puede entrar una línea de comando del siguiente formato:

C N1, N2, S1, S2 seguido por **ENTER**

C es el comando a ejecutar (ver "Los comandos del Editor").
N1 es un número en el rango 1 - 32767 inclusive.
N2 es un número en el rango 1 - 32767 inclusive.
S1 es una cadena de caracteres con una longitud máxima de 20.
S2 es una cadena de caracteres con una longitud máxima de 20.

La coma se utiliza para separar los diversos argumentos (aunque esta puede cambiarse - ver el comando '**S**') y los espacios se ignoran, excepto dentro de las cadenas. Ninguno de los argumentos es obligatorio, aunque alguno de los comandos (p.e. el comando '**D**'elete) no funcionará si no se especifican *N1* y *N2*. El editor recuerda los números y cadenas previos que entró y utiliza estos valores anteriores, donde proceda, si no especifica un argumento particular en la línea de comandos. Los valores de *N1* y *N2* estan asignados inicialmente a diez y las cadenas están inicialmente vacías. Si entra una línea de comandos ilegal como **F-1,100,HOLA** , la línea se ignora y se muestra el mensaje '*Pardon?*' - deberá reescribir la línea correctamente p.e. **F1,100,HOLA**. Este mensaje de error también se muestra si la longitud de *S2* excede de 20; si la longitud de *S1* es mayor de 20, los caracteres sobrantes se ignoran.

Los comandos pueden entrarse en modo mayúsculas o minúsculas.

Mientras se entra un línea de comandos, se pueden utilizar todas las funciones de control pertinentes descritas anteriormente p.e. **CX** para borrar hasta el principio de la línea, **CI** para avanzar el cursor a la siguiente posición *tab*, etc.

La siguiente subsección detalla los diversos comandos disponibles en el editor - observe que siempre que un argumento va encerrado entre los símbolos '<>', este argumento debe estar presente para que el comando trabaje.

Los comandos del Editor.

Inserción de texto.

Se puede insertar texto en el fichero ya sea escribiendo un número de línea, un espacio y luego el texto requerido o utilizando el comando '**I**'. Observe que si escribe un número de línea seguido de ENTER. (es decir, sin ningún texto), esa línea

será borrada del texto si esta existe. Siempre que se está entrando texto, se pueden emplear las funciones de control **CX** (borra hasta el principio de la línea), **CI** (va a la siguiente posición TAB) y **CC** (vuelve al bucle central de comandos). La tecla DELETE (**CH**) provocará un borrado (pero no hasta el principio de la línea de texto). El texto es entrado en un buffer interno de GENS3, y si este buffer se llenase, se te prohibiría seguir entrando más texto - debe entonces usar **CH** o **CX** para liberar espacio en el buffer.

Si, durante la inserción de texto, el editor detecta que el final del texto está cerca del tope de la RAM, muestra el mensaje '*Bad Memory!*'. Esto indica que no se puede insertar más texto y que el fichero de texto actual, o como mínimo una parte de ella, debe salvarse en cinta para una posterior recuperación.

Comando: I n,m

Este comando hace que GENS3 vaya indicando números de línea de forma automática: se le apuntarán números de línea empezando en **n** e incrementando en intervalos de **m**. Entre el texto oportuno después del número de línea mostrado, utilizando los diversos códigos de control si deseas, y terminando la línea de texto con **ENTER**. Para salir de este modo utilice la función de control **CC**.

Si entra una línea con un número de línea que ya existe en el texto, la línea existente será borrada y reemplazada por la nueva línea, después de que haya pulsado **ENTER**. Si el incremento automático del número de línea produce una línea mayor de 32767, se saldrá automáticamente del modo Inserción.

Si, cuando escribe un texto, llega al final de la línea de pantalla sin haber entrado 64 caracteres (el tamaño del buffer), la pantalla se 'moverá' (scroll) hacia arriba y podrá continuar escribiendo en la siguiente línea - se dará un escalonamiento automático al texto ya que los números de línea están efectivamente separados del texto.

Listado del texto.

El texto puede inspeccionarse utilizando el comando '**L**'; el número de líneas mostrado en todo momento durante la ejecución de este comando es fijado inicialmente, pero puede cambiarse utilizando el comando '**K**'.

Comando: L n,m

Lista el texto actual en el dispositivo adecuado desde el número de linea **n** hasta el número **m** inclusive. El valor por defecto de

n es siempre 1 y el valor por defecto para m es siempre 32767, es decir, los valores por defecto no son cogidos desde argumentos previamente entrados. Para listar el fichero de texto completo, utilice simplemente '**L**' sin ningún argumento. Las líneas de pantalla son formateadas con un margen izquierdo para que el número de línea se muestre claramente. La tabulación de la línea es automática, resultando en una clara separación de los diversos campos de la línea. El número de líneas de pantalla listados en el elemento escogido puede controlarse utilizando el comando '**K**' - después de listar cierto número de líneas, el listado se detendrá momentáneamente (si no ha llegado ya al número de línea **m**), pulse la función de control **CC** para volver al bucle central del editor o cualquier otra tecla para continuar el listado.

Comando: K n

'**K**' asigna el número de líneas a listar en el elemento adecuado antes de que el listado se detenga como se describe en '**L**'. El valor (*n MOD 256*) se calcula y se almacena. Por ejemplo, use **K5** si desea un '***L****'istado* posterior que produzca 5 líneas a la vez.

Edición de texto.

Una vez se ha creado algún texto, habrá una inevitable necesidad de editar algunas líneas. Existen varios comandos para hacer posible la corrección, borrado, movimiento y renumeración de líneas:

Comando: D <n,m>

Todas la líneas desde **n** hasta **m** inclusive, se borran del fichero de texto. Si **m<n**, o se especifican menos de dos argumentos, no se realizará ninguna acción; esto se hace para prevenir errores por descuido. Se puede borrar una simple línea haciendo **m=n**; también se puede hacer esto tecleando simplemente el número de línea seguido por **ENTER**.

Comando M n,m

Esto provoca que el texto de la línea **n** sea entrado en la línea **m**, borrando cualquier texto que existiera en esta. Observe que la línea **n** queda igual. Por tanto, este comando le permite '***M****'over* una línea de texto a otra posición del fichero. Si el número de línea **n** no existe, no se producirá ninguna acción.

Comando: N <n,m>

La utilización del comando '**N**' provoca que el fichero de texto sea renumerado con un primer número de línea igual a **n** y con un intervalo entre líneas de **m**. Ambos, **n** y **m** deben estar presentes, y si la renumeración produjese algún número de línea que excediese de 32767, se dejaría la numeración original.

Comando: F n,m,f,s

El texto existente en el rango de líneas de **n < x < m** se explora para encontrar la cadena **f**, la cadena "a encontrar". Si se encontrase la cadena, se mostraría la línea pertinente y se entraría en modo Edición (verlo más adelante). Puede entonces utilizar los comandos del modo Edición para buscar la cadena en las líneas siguientes dentro del rango de líneas definido, o sustituir la cadena **s** (la cadena "substituta") por la actual **f** y luego buscar la siguiente localización de **f**; ver más adelante para más detalles.

Observe que el rango de líneas y las dos cadenas pueden haber sido previamente asignadas por otro comando, por tanto tan solo puede ser necesario entrar '**F**' para iniciar la búsqueda - ver el ejemplo más adelante para clarificar.

Comando: E n

Edita la línea con número de línea **n**. Si **n** no existe no se producirá ninguna acción; en caso contrario, la línea se copia en un buffer y se muestra en la pantalla (con el número de línea), el número de línea se muestra de nuevo bajo la línea y se entra en modo Edición. Todas las ediciones siguientes tendrán lugar en el buffer y no en el mismo texto; así, la línea original puede recuperarse en cualquier momento.

De este modo nos imaginamos un puntero moviéndose a través de la línea (empezando en el primer caracter) y varios sub-comandos que le permiten editar la línea. Los subcomandos son:

' ' (Espacio) - Incrementa el puntero de texto en uno, es decir, apunta al siguiente caracter de la línea. No puede ir más allá del final de la línea.

CH (DELETE) - Decremento el puntero de texto en uno para apuntar al caracter anterior de la línea. No puede retroceder más allá del principio de la línea.

CI (Función de Control) - Avanza el puntero de texto hasta la siguiente posición de tabulación de cada linea de pantalla.

ENTER - Finaliza la edición de esta línea, guardando todos los cambios realizados.

Q - Deja la edición de esta línea, es decir, abandona la edición ignorando todos los cambios realizados y dejando la línea como estaba antes de iniciar la edición.
R - Re-carga el buffer de edición desde el texto, es decir, olvida todos los cambios hechos en esta línea y la restituye como era originalmente.

L - Lista el resto de la línea editada, es decir, el resto de la línea más allá de la posición actual del puntero. Seguirá en modo Edición con el puntero reposicionado al principio de la línea.

K - Elimina (borra) el caracter de la posición actual del puntero.

Z - Borra todos los caracteres desde (e incluyendo) la posición actual del puntero hasta el final de la línea.

F - Localiza la siguiente cadena "a encontrar" previamente definida en una línea de comando (ver comando '**F**'). Este subcomando saldrá automáticamente de la edición de la línea en curso (manteniendo los cambios) si no encuentra otra cadena "a encontrar" en la línea actual. Si se detecta la cadena en una línea posterior (dentro del rango previamente especificado), se entrará en modo Edición para la línea en que se encontró la cadena. Observe que el puntero está siempre posicionado al principio de la cadena localizada.

S - Substituye la cadena "substituta" previamente definida en la posición actual de la cadena "a encontrar" y después ejecuta el subcomando '**F**', es decir, busca la siguiente posición de la cadena "a encontrar". Esto, junto con el anterior. subcomando '**F**' se utiliza para moverse a través del fichero de texto, reemplazando opcionalmente las apariciones de la cadena "a encontrar" por la cadena "substituta" - ver ejemplo más adelante.

I - Inserta caracteres en la posición actual del puntero. Permanecerás en este submodo hasta que pulses **ENTER** - esto volverá al modo principal Edición con el puntero posicionado después del último caracter insertado. Utilizando **CH** (DELETE) en este submodo, provocará el borrado del caracter a la izquierda del puntero, mientras que el uso de **CI** (Función de Control) avanzará el puntero a la siguiente posición de tabulación, insertando espacios.

X - Avanza el puntero al final de la línea y automáticamente entra el submodo de inserción detallado antes.

C - Submodo de cambio. Esto le permite sobreescribir el caracter de la posición actual del apuntador y luego avanza el puntero en uno. Permanecerá en el submodo de cambio hasta que pulse **ENTER** donde será llevado al modo Edición con el puntero colocado después del último caracter que cambió. **CH**(DELETE) en este submodo decremento el apuntador en uno, es decir, lo mueve a la izquierda, mientras **CI** no tiene efecto.

Comandos de cinta.

El texto puede salvarse o cargarse desde cinta utilizando los comandos '**P**', '**G**' y '**T**'.

Comando: P n,m,s

El rango de líneas definido por **n < x < m** se salva en cinta bajo el nombre de fichero especificado por la cadena **s**. Recuerde que estos argumentos pueden haber sido asignados por un comando previo. Antes de entrar este comando, asegúrese de que su cassette está conectado y en modo *RECORD* (grabación). No utilice este comando si desea, en un paso posterior, 'Incluir' el texto. Utilice en su lugar el comando '**T**'.

Comando: G,,s

Se explora la cinta para buscar un fichero con un nombre de **s**. Cuando se encuentra, se carga al final del texto actual. Si se indica una cadena nula como nombre de fichero, se cargará el primer fichero que se encuentre.

Después de que haya entrado el comando '**G**', se muestra el mensaje '*Start tape..*' - ahora deberá pulsar *PLAY* en su reproductor de cinta. Se buscará un fichero de texto con el nombre especificado, o el primer fichero de texto si se dio un nombre nulo. Si se encuentra, se mostrará el mensaje '*Using nombre*', en caso contrario se mostrará '*Found nombre*' y proseguirá la exploración de la cinta.

Observe que si algún fichero de texto ya está presente en memoria, el fichero cargado desde la cinta se añadirá al existente y el fichero completo se renumerará empezando con la línea 1 en pasos de 1.

Comando: T n,m,s

Salva un bloque de texto, entre los números de línea **n** y **m** inclusive, grabándolo en un formato adecuado para su inclusión en un momento posterior por medio del comando del ensamblador ***F** El fichero se salva con el nombre de **s**. El *SAVE* tiene lugar inmediatamente después de pulsar *ENTER* por tanto, deberá asegurarse de que su cassette está listo para grabar antes de entrar esta línea de comando.

Observe que no deberá utilizar este comando si lo que desea es añadir el texto posteriormente, como tampoco deberá utilizar el comando '**P**' si desea 'incluir' el texto.

Ensamblando y ejecutando desde el Editor.

Comando: A

Hace que el texto se ensamble desde la primera línea del fichero de texto.

Comando: R

Si el fuente se ha ensamblado sin errores y se ha especificado una dirección de ejecución por medio del pseudo-nemónico *ENT*, puede utilizarse el comando '**R**' para ejecutar el programa objeto. El programa objeto puede utilizar una instrucción *RET* (C9) para volver al editor tan pronto como la pila esté en la misma posición al final de la ejecución en que estaba al principio. Observe que *ENT* no tendrá efecto si se ha especificado la *Opción 16* para el ensamblado.

Otros comandos.

Comando: B

Simplemente devuelve el control al sistema operativo. Para reentrar al ensamblador, utilice bien una entrada "en frío", o "en caliente".

Comando: C

Este comando le permite convertir ficheros de texto producidos por GENS1 al formato de texto comprimido de GENS3. Simplemente cargue el fichero de texto de GENS1, utilizando el comando '**G**', utilice el comando '**C**' para convertir el fichero y luego salve el fichero comprimido utilizando el comando '**P**'.

'**C**' no toma argumentos y puede tomar un tiempo sustancial para completar la conversión del fichero.

Vea modificaciones para la última versión.

Comando: S,,d

Este comando le permite cambiar el delimitador que se toma como separación de los argumentos en la línea de comandos. Al entrar en el Editor se toma la coma ',' como delimitador; esto puede cambiarse utilizando el comando '**S**' al primer caracter de la cadena **d**. Recuerde que una vez ha definido un nuevo delimitador, debe utilizarse (incluso en el comando '**S**') hasta que se especifique otro.

Observe que el separador no puede ser un espacio.

Comando: V

El comando '**V**' muestra los valores actuales de *N1, N2, S1* y *S2*, es decir, los dos números de línea por defecto y las dos cadenas. Esto es útil antes de entrar algún comando en que va a utilizar los valores por defecto, para ver si estos valores son correctos.

Comando: W n,m

El comando '**W**' provoca que la sección de texto entre las líneas **n** y **m** inclusive sean sacadas por impresora. Si ambas **n** y **m** se omiten, se imprimirá el fichero completo. La impresión se detendrá momentáneamente después del número de líneas asignado por el comando '**K**' - pulse alguna tecla para continuar la impresión.

Comando: X

'**X**' simplemente provoca que las direcciones de principio y final del fichero de texto, se muestren en decimal. Es útil si desea salvar el texto desde Basic, o si quiere ver cuanta memoria ha dejado después del fichero de texto. GENS3 siempre supone que el texto empieza en la primera posición dada por el comando '**X**' y guarda la dirección final del texto en la posición *TEXEND* la cual está en '*principio de GENS3 + 54*'. Así, si desea modificar un fichero de texto (quizá producido por MONS3), debe mover el fichero de texto a la dirección especificada por la primera dirección mostrada por el comando '**X**', modifique *TEXTEND* para que contenga la dirección final del fichero y finalmente entre GENS3 por medio de una "entrada en frío". Por ejemplo, digamos que ha generado un fichero de texto en la situación correcta, y que acaba (la dirección después del indicador de final de linea) en ***#9A02***. Entonces, asumiendo que ha cargado GENS3 en **24064**, debería hacer desde *BASIC*, **POKE 24064+54,2** (#**02**) y **POKE 24064+55,154** (#**9A**) y luego entre en el GENS3 con **RANDOMIZE USR 24125**. Ahora podrá trabajar con el fichero de texto normalmente desde el editor.

Un ejemplo de utilización del Editor.

Asumamos que ha escrita el siguiente programa (usando I10,10):

```
10 *h     NUMEROS ALEATORIOS DE 16 BITS
20
30 ;ENTRADA:  HL contiene los números random previos.
40 ;SALIDA:   HL contiene los nuevos números randon.
50
60 Random  PUSH AF        ;salva los registros
70 PUSH BC
80         PUSH HL
90         ADD HL,HL      ;*2
100        ADD HL,HL      ;*4
110        ADD HL,HL      ;*8
120        ADD HL,HL      ;*16
130        ADD HL,HL      ;*32
140        ADD HL,HL      ;*64
150        PIP 8C         ;número aleatorio antiguo
160        ADD HL,DE
170        LD DE,41
180        ADD HL,DE
190        POP BC         ;restaura registros
200        POP AF
210        REY
```

Este programa tiene varios errores:

Línea 10: se ha utilizado una minúscula en el comando *H.
Línea 40: '*randon*' en lugar de '*random*'.
Línea 70: PUSH BC empieza en el campo de etiquetas.
Línea 150: '*PIP*' en lugar de '*POP*'.
Línea 160: necesita un comentario (no es un error - simplemente un estilo).
Línea 210: '*REY*' debería ser '*RET*'.

También deberán añadirse dos líneas extra de *ADD HL,HL* entre las líneas 140 y 150 y todas las referencias al par de registros *DE* de las líneas 160 a 180 deberían ser al par de registros BC.

Para poner todo esto bien, podemos proceder como sigue:

E10 ENTER luego _(espacio) **C** (modo de cambio) **H <ENTER> <ENTER>**
F40,40,randon,random _(espacio) después el subcomando '**S**'.
E70 ENTER luego **I**(Inserción)**-------**(7 espacios) **<ENTER> <ENTER>**
I142,2 ENTER 142_____(8 espacios)**ADD HL,HL   ;*128**
144   ADD HL,HL   ;*256 (y **CC** para salir).
F150,150,PIP,POP <ENTER> luego el subcomando '**S**'.
E160 ENTER luego **X__;*257 + 41 <ENTER> <ENTER>**
F160,180,DE,BC <ENTER> luego uso repetido del subcomando '**S**'.
E210 ENTER CI CI___C (modo cambio) **T <ENTER> <ENTER>**
N10,10 <ENTER> para enumerar el texto.

Le recomendamos trabaje este ejemplo utilizando el editor.

APÉNDICE 1 - NÚMEROS DE ERROR Y SUS SIGNIFICADOS.

***ERROR* 1** Error en el contexto de esta línea.

***ERROR* 2** Nemónico no reconocido.

***ERROR* 3** Instrucción mal formada.

***ERROR* 4** Símbolo definido más de una vez.

***ERROR* 5** Esta línea contiene un caracter ilegal,es decir, un caracter que no es válido en un contexto particular.

***ERROR* 6** Uno de los operandos de esta línea es ilegal.

***ERROR* 7** Un símbolo de esta línea es una palabra reservada.

***ERROR* 8** Desproporción de registros.

***ERROR* 9** Demasiados registros en esta línea.

***ERROR* 10** Una expresión que debería evaluar 8 bits, evalúa más de 8 bits.

***ERROR* 11** Las instrucciones JP (IX+n) y JP (IY+n) son ilegales.

***ERROR* 12** Error en la información de un pseudo-nemónico.

***ERROR* 13** Referencia adelantada ilegal, es decir, se ha hecho un EQU a un símbolo que todavía no ha sido definido.

***ERROR* 14** División por cero.

***ERROR* 15** Overflow en una operación de multiplicación.

Bad ORG! Se ha hecho un ORG a una dirección que afectaría a GENS3, su fichero de texto o la tabla de símbolos. El control vuelve al Editor.

Out of Tabla space! Se produce durante la primera pasada si se ha asignado una memoria insuficiente para la Tabla de Símbolos. El control vuelve inmediatamente al Editor.

Bad Memory! Se muestra si no hay espacio para insertar más texto, es decir, el final del texto está cerca del tope de RAM. Deberá salvar el texto, o parte de él, en cinta.

APÉNDICE 2 - PALABRAS RESERVADAS, NEMONICOS ETC.

Lo siguiente es una lista de las palabras reservadas de GENS3. Estos símbolos no pueden utilizarse como etiquetas, aunque pueden formar parte de una. Observe que todas las palabras reservadas se componen de letras mayúsculas.

A	**B**	**C**	**D**	**E**	**H**	**L**	**I**	**R**	**$**
AF		**AF**		**BC**		**DE**		**HL**	**IX**
IY		**SP**		**NC**		**Z**		**NZ**	**M**
P		**PE**		**PO**					

Ahora sigue una lista de los nemónicos válidos del Z80, pseudo-nemónicos y comandos ensamblador. Observe que éstos también también deben entrarse en letras mayúsculas.

ADC	**ADD**	**AND**	**BIT**	**CALL**	**CCF**	**CP**	**CPD**	**CPDR**
CPI	**CPIR**	**CPL**	**DAA**	**DEC**	**DI**	**DJNZ**	**EI**	**EX**
EXX	**HALT**	**IM**	**IN**	**INC**	**IND**	**INDR**	**INI**	**INIR**
JP	**JR**	**LD**	**LDD**	**LDDR**	**LDI**	**LDIR**	**NEG**	**NOP**
OR	**OTDR**	**OTIR**	**OUT**	**OUTD**	**OUTI**	**POP**	**PUSH**	**RES**
RET	**RETI**	**RETN**	**RL**	**RLA**	**RLC**	**RUCA**	**RLD**	**RR**
RRA	**RRC**	**RRCA**	**RRD**	**RST**	**SBC**	**SCF**	**SET**	**SLA**
SRA	**SRL**	**SUB**	**XOR**					

DEFB	**DEFM**	**DEFS**	**DEFW**	**ELSE**	**ENO**	**ENT**	**EQU**	**IF**
ORG								

***D**	***E**	***H**	***L**	***S**	***C**	***F**

APÉNDICE 3 - UN EJEMPLO TRABAJADO.

Aquí sigue un ejemplo de una sesión típica utilizando GENS3 - si es un principiante en el mundo de los programas en ensamblador o si simplemente está un poco inseguro de como utilizar el editor/ ensamblador, le instamos a que trabaje este ejemplo cuidadosamente. Observe que se utiliza **<ENTER>** para indicar que debe pulsar **ENTER** en el teclado.

Objetivo de la sesión:

Escribir y probar una rutina rápida de multiplicación entera, el texto de la cual será salvado en cinta usando el comando '**T**' del editor para que pueda incluirse fácilmente en futuros programas. Plan de trabajo de la sesión:

1. Escribir la rutina como una subrutina y salvarla en cinta utilizando el comando '**P**' del editor para que pueda ser fácilmente recuperada y editada durante esta sesión, si hubiese algún error.

2. Depurar la rutina de multiplicación, editándola como se requiera.

3. Salvar la rutina depurada en cinta, utilizando el comando '**T**' para que la rutina pueda incluirse en otros programas.

Antes de que empezar, debemos cargar GENS3 en el ordenador - haga esto escribiendo **LOAD "" CODE 24064** para cargar el ensamblador en la dirección **24064**. Ahora escriba **RANDOMIZE USR 24064**, y en respuesta a la pregunta '*Buffer size?*' escriba **1** seguido por **<ENTER>** para crear un buffer '*Include*' de un tamaño de 1*256 bytes. Ahora aparecerá en pantalla un signo '**>**' - está en modo editor listo para crear programas en ensamblador.

Paso 1 - escribir la rutina de multiplicación entera.

Usamos el comando '**I**' del editor para insertar el texto, utilizando **CI** (el caracter de tabulación) para obtener un listado tabulado. No necesitamos utilizar **CI**, un listado del texto siempre hará la tabulación por nosotros. No hemos indicado donde se ha utilizado **CI** pero puede asumir que han sido utilizados antes del nemónico y entre el nemónico y el operando. Observe que las direcciones mostradas en los listados ensambladores del ejemplo pueden no corresponder a las producidas por su máquina; éstas sirven solamente como propósito ilustrativo.

```
>I10,10 <ENTER>
10 ;rutina rápida de multiplicación  <ENTER>
20 ;entera. Multiplica HL por  <ENTER>
30 ;DE. Devuelve el resultado  <ENTER>
40 ;en HL. El banderín C se activa  <ENTER>
50 ;si se produce overflow.  <ENTER>
60 <ENTER>
70         ORG   #7F00  <ENTER>
80 <ENTER>
90 Mult    OR    A  ENTER>
100        SBC   HL,DE ;HL>DE?  <ENTER>
110        ADD   HL,DE  <ENTER>
120        JR    NC,Mu1 ;sí  <ENTER>
130        EX    DE,HL  <ENTER>
140 Mul    OR    D  <ENTER>
150        SCF ; overflow si  <ENTER>
160        RET   NZ ;DE>255  <ENTER>
170        OR    E ;veces 0?  <ENTER>
180        LD    E,0  <ENTER>
190        JR    NZ,Mu4 ;no  <ENTER>
200        EX    DE,HL ;0  <ENTER>
210        RET  <ENTER>
220 <ENTER>
230 ;Rutina principal.  <ENTER>
240 <ENTER>
250 Mu2    EX    DE,HL  <ENTER>
260        ADD   HL,DE  <ENTER>
270        EX    DE,HL  <ENTER>
280 Mu3    ADD   HL,HL  <ENTER>
290        RET   C ;OVERFLOW  <ENTER>
300 Mu4    RRA  <ENTER>
310        JR    NC,Mu3  <ENTER>
320        OR    A  <ENTER>
330        JR    NZ,Mu2  <ENTER>
340        ADD   HL,DE  <ENTER>
350        RET  <ENTER>
360 CC
>P10,350,Mult  <ENTER>
```

Lo anterior creará el texto de la rutina y lo salvará en cinta. Recuerde que ha de tener su cassette en marcha y en modo *RECORD* antes antes de ejecutar el comando '**P**'.

Paso 2 - Depurando la rutina.

Primero, vamos ver si el texto ensambla correctamente. Utilizaremos la *Opción 6* para que no se produzca ningún listada ni se genere ningún código objeto.

```
>A  <ENTER>
Table size: <ENTER>(valor por defecto para la tabla de símbolos)
Options: 6  <ENTER>
```

```
*HISOFT GENS3 ASSEMBLER*
 Copyright Hisoft 1983
  All Rights Reserved

Pass 1 errors: 00

Pass 2 errors: 00

*WARNING* MU4 absent
Table used:   74 from 161
>
```

Vemos por el ensamblado que hemos tenido un error en la línea 190, ya que hemos entrado MU4 en lugar de Mu4, que es la etiqueta a la que deseamos saltar. Así, edita la linea 190:

```
>F190,190,MU4,Mu4 <ENTER>
  190      JR   NZ, (ahora utilice el subcomando 'S')
```

Ahora ensamble el texto de nuevo y encontrará que ensambla sin errores. Ahora debemos escribir algún código para probar la rutina:

```
>N300,10 <ENTER> (renumera para poder escribir más texto)
>I10,10 <ENTER>
 10 ;Algo de código para probar <ENTER>
 20 ;La rutina Mult. <ENTER>
 30 <ENTER>
 40         LD    HL,50 <ENTER>
 50         LD    DE,20 <ENTER>
 60         CALL, Mult ;Multiplica <ENTER>
 70         LD    A,H ;resultado de la operación <ENTER>
 80         CALL  Aout <ENTER>
 90         LD    A,L <ENTER>
100         CALL  Aout <ENTER>
110         RET ; vuelve al editor <ENTER>
120 <ENTER>
130 ;Rutina para sacar A en hexadecimal <ENTER>
140 <ENTER>
150 Aout    PUSH  AF <ENTER>
160         RRCA <ENTER>
170         RRCA <ENTER>
180         RRCA <ENTER>
190         RECA <ENTER>
200         CALL  Nibble <ENTER>
210         POP   AF <ENTER>
220 Nibble  AND   %1111 <ENTER>
230         ADD   A,#90 <ENTER>
240         DAA <ENTER>
250         ADC   A,#40 <ENTER>
260         DAA <ENTER>
270         LD    IY,#5C3A ;para ROM <ENTER>
280         RST   #10; llamada ROM <ENTER>
290         RET <ENTER>
300 CC
>
```

Ahora ensamble la rutina de prueba y la rutina de multiplicación juntas.

>**A**

Table size: <ENTER>
Options: **6** <ENTER>

HISOFT GENS3 ASSEMBLER
 Copyright HISOFT 1983
  All rights reserved

7EAC 190   RECA
**ERROR* 02* (Pulse alguna tecla para continuar)

Pass 1 errors: 01
Table used: 88 from 210
>

Tenemos un error en nuestra rutina ;RECA debería ser RRCA en la línea 190. Así:

>**E190**
  190   **RECA**
  190   **C**(entra modo cambio)**R** <ENTER> <ENTER>

Ahora ensamble de nuevo, utilizando simplemente la *Opción 4* (sin listado), y el texto ensamblará correctamente Asumiendo que lo hace, ahora estamos en posición de probar el trabajo de nuestra rutina Mult, por lo tanto necesitamos decirle al editor desde dónde puede ejecutar el código. Hacemos esto con el pseudo-nemónico ENT:

>**35 ENT $ <ENTER>**

Ahora ensamble el texto de nuevo y el ensamblado terminará correctamente con los mensajes:

Table used: 88 from 211
Executes: 32416

o algo parecido. Ahora podemos ejecutar nuestro código utilizando el comando '**R**' del editor. Suponemos que multiplicará 50 por 20 produciendo 1000 que es 3E8 en hexadecimal.

>R <ENTER>
0032>

¡No trabaja! ¿Por qué no? Liste las líneas 380 a 500 (**L380,500**). Verá que en la línea 430 la instrucción es un *OR D* seguida, efectivamente, por un RET NZ. Lo que esto está haciendo es un *OR* lógico entre el registro *D* y el acumulador *A* y volviendo con un banderín de error activado (banderín C) si el resultado es distinto de cero. El objeto de esto es asegurar que *DE<256* para que la multiplicación no produzca *Overflow* - hace esto comprobando que *D* es cero... pero el *OR* solo funcionará

correctamente en este caso si el acumulador *A* es cero para empezar, y no tenemos garantías de que esto sea así. Debemos asegurarnos de que *A* es cero antes de hacer el *OR D*, en otro caso obtendremos un Overflow impredecible con el número más alto devuelto como resultado. Inspeccionando el código, vemos que el *OR A* de la línea 380 puede hacerse en un *XOR A* activando así los banderines para la instrucción *SBC HL,DE* y asignando *0* a *A*. Así:

>**E380 <ENTER>**
380 Mult OR A
380 I (entra molo Inserción) **X <ENTER> 4 <ENTER>**

Ahora ensamble de nuevo (*Opción 4*) y ejecute el código, utilizando '**R**'. La respuesta ahora sería correcta - 3E8.

Podemos hacer más pruebas de la rutina editando las líneas 40 y 50 para multiplicar diferentes números y después ensamblar y ejecutar - encontrará que la rutina trabaja perfectamente.

Ahora que hemos perfeccionado la rutina, podemos salvarla en cinta en formato '*Include*':

>**T300,999,Mult <ENTER>**

Recuerde colocar el cassette en modo RECORD (grabación) antes de pulsar ENTER. Una vez la rutina ha sido salvada como está, puede incluirse en otro programa como se muestra:

```
500       RET
510
520 ;Incluye la rutina de multiplicación aquí.
530
540 *F Mult
550
560 ;La siguiente rutina.
```

Cuando el texto anterior se ensambla, el ensamblador le sugerirá '*Start tape..*' cuando llegue a la línea 540 en el primer y segundo paso. Por lo tanto debería tener la rutina *Mult* salvada dos veces en la cinta. Esto normalmente significa rebobinar la cinta después del primer paso. Puede grabar dos veces *Mult* en la cinta, siguiendo una a la otra, y utilizar una para el primer paso y la otra para el segundo.

Por favor, estudie el ejemplo anterior cuidadosamente y pruebe a hacerlo usted mismo.


MONS 3

PUESTA A PUNTO.

MONS3 es un programa totalmente relocalizable; simplemente cárguelo en la dirección desde la que desea que se ejecute y luego entre en MONS3 por medio de esa dirección. Si desea entrar en MONS3 de nuevo (habiendo vuelto desde MONS3 a BASIC), ejecútelo a través de una dirección mayor en 29 (decimal) que la dirección original.

Ejemplo:

Digamos que quiere cargar MONS3 en la dirección #C000 (49152 decimal) - proceda como sigue:

LOAD "" CODE 49152
RANDOMIZE USR 49152

Para entrar MONS3 de nuevo, utilice RANDOMIZE USR 49181 - esto evita la sección de código que reasigna las direcciones de MONS3 en la primera entrada.

MONS3 es de unos 5K de longitud una vez ha sido relocalizado, pero debe dejar cerca de 6K bytes al cargarlo debido a la tabla de asignación de direcciones que viene después del código principal. MONS3 contiene su propia pila interna.

Una vez ha entrado MONS3, el mensaje '**MONS3 Copyright Hisoft 1983**' aparecerá por unos pocos segundos y será reemplazado por un 'panel frontal' (Ver Apéndice para un ejemplo de pantalla). Este consta de los registros del Z80 y los banderines junto con su contenido más una sección de 24 bytes de memoria centrada (utilizando '**<**' y '**>**') alrededor del valor actual del Apuntador de Memoria que inicialmente está asignado a la posición 0. En la línea superior de la pantalla hay un desensamblado de la instrucción direccionada por el Apuntador de Memoria.

Al entrar en MONS3, todas las direcciones mostradas en el panel se dan en formato hexadecimal (es decir, en base 16); puede cambiar esto para que las direcciones se muestren en decimal utilizando el comando **SYMBOL SHIFT 3**. Notará, sin embargo, que las direcciones siempre deben entrarse en hexadecimal.

Los comandos se entran desde teclado en respuesta al indicador '**>**' situado bajo el display de la memoria y pueden entrarse en mayúsculas o minúsculas. Algunos comandos, cuyo efecto podría ser desastroso si se utilizan por error, requieren que pulse *SYMBOL SHIFT* así como la letra del comando. A lo largo de este manual la utilización de la tecla *SYMBOL SHIFT* será representada por el símbolo "**^**". p.e. **^Z** significa pulsar *SYMBOL SHIFT y Z* juntas.

Los comandos tienen efecto inmediatamente - no hay necesidad de terminarlos con <ENTER>. Los comandos no válidos simplemente se ignoran. El panel completo se actualiza después de que cada comando se procese para que pueda observar algunos resultados del comando particular.

Muchos comandos requieren la entrada de un número hexadecimal. Cuando entre un número hexadecimal podrá entrar cuantos dígitos hexadecimales (0-9 y A-F o a-f) desee y terminar con cualquier dígito no hexadecimal. Si el terminador es un comando válido, el comando se asume después de que algún comando previo haya sido procesado. Si el terminador es un signo menos '**-**' se devuelve el negativo del número hexadecimal entrado - en forma de complemento a dos p.e. 1800- da E800. Si entra más de cuatro dígitos cuando escriba un número hexadecimal, sólo se retienen y muestran en la pantalla los cuatro últimos tecleados.

Si, en cualquier momento, desea volver al intérprete BASIC desde MONS3, pulse simplemente **CAPS SHIFT 1**.

NOTA IMPORTANTE MONS3 inhabilita las interrupciones para asegurar su correcto funcionamiento. El usuario debe asegurarse de que las interrupciones no se rehabilitan durante un sesión con MONS3.

LOS COMANDOS DISPONIBLES.

En MONS3 están disponibles los siguientes comandos. En esta sección, siempre que se utiliza <ENTER> para terminar un número hexadecimal, puede de hecho, ser cualquier caracter no hexadecimal. También '**-**' se utiliza para denotar un espacio donde convenga.

SYMBOL SHIFT 3

Mueve la base en la cual se muestran las direcciones, entre la base 16 (hexadecimal) y la base 10 (decimal). Al entrar en MONS3 las direcciones se muestran en hexadecimal, utilice **^3** para cambiar a un formato decimal y **^3** de nuevo, para regresar al formato hexadecimal. Esto afecta a todas las direcciones mostradas por MONS3 incluyendo aquellas generadas por el desensamblador, pero no cambia la pantalla del contenido de memoria - ésta siempre se da en hexadecimal.

SYMBOL SHIFT 4 ó '$'

Muestra una página de desensamblado empezando en la dirección almacenada por el Apuntador de Memoria. Útil para mirar delante de la posición actual para ver qué instrucciones vienen. Pulse **^4** de nuevo para volver al 'panel frontal' o cualquier otra tecla para obtener una página posterior de desensamblado.

ENTER

Incrementa el Puntero de Memoria en uno para que la pantalla de 24 bytes de memoria se centre alrededor de una dirección mayor en uno de lo que era previamente.

CAPS SHIFT 7

Decrementa el Puntero de Memoria en uno.

CAPS SHIFT 5

Decrementa el apuntador de memoria en ocho. Se usa para retroceder rápidamente.

CAPS SHIFT 8

Incrementa el Puntero de Memoria en ocho. Se usa para avanzar rápidamente.

',' (Coma)

Actualiza el Puntero de Memoria para que contenga la dirección actual de la pila (indicada por SP). Esto es útil cuando quiere mirar la dirección de vuelta de una rutina llamada, etc.

'G'

Busca en la memoria una cadena especifica.

Aparece en pantalla un ':' y deberá entrar el primer byte que quiere que busque seguido por <ENTER> - ahora entre los siguientes bytes (y <ENTER>) en respuesta al ':' hasta que haya definido la cadena completa. Entonces pulse solo <ENTER> en respuesta al ':'; esto terminará la definición de la cadena y comenzará la búsqueda en la memoria, empezando desde la actual dirección del Puntero de Memoria, de la primera aparición de la cadena especificada. Cuando se encuentra la cadena, el 'panel frontal' se actualiza para que el Puntero de Memoria se posicione en el primer caracter de la cadena. Ejemplo:

Digamos que desea buscar en memoria, empezando desde #8000, el caracter #3E #FF (2 bytes) - proceda coma sigue:

M:8000 <ENTER>	Asigna #8000 al Puntero de Memoria.
G:3E <ENTER>	Define el primer byte de la cadena.
FF <ENTER>	Define el segundo byte de la cadena.
<ENTER>	Termina la cadena.

Después del ENTER final (o cualquier caracter no hexadecimal) '**G**' procederá a buscar en la memoria desde #8000, el caracter #3E #FF. Cuando se encuentra, el display se actualiza - para buscar posteriores localizaciones de la cadena, utilice el comando '**N**'.

'H'

Convierte un número decimal a su equivalente hexadecimal.

Aparece en pantalla el signo ':' para entrar un número decimal terminado por algún caracter no-dígito (es decir un carácter

distinto de 0. .9 inclusive). Una vez se ha terminado el número, se muestra un signo '**=**' en la misma línea seguido por el equivalente hexadecimal del número decimal. Ahora pulse alguna tecla para volver al modo comando. Ejemplo:

H:41472_=A200 aquí se ha utilizado un espacio como terminador.

'I'

Copia inteligente.

Se utiliza para copiar un bloque de memoria desde una posición a otra. Es inteligente en que el bloque de memoria puede ser copiado en posiciones donde arrollaría sus posiciones previas.

'**I**' pide las direcciones de principio y final inclusives del bloque a copiar ('*First:*','*Last:*') y luego la dirección a la cual se va a mover el bloque ('*To:*'); entre números hexadecimales en respuesta a cada una de estas preguntas. Si la posición inicial es mayor que la posición final, se anula el comando, en caso contrario, el bloque se mueve como se ordenó.

'J'

Ejecuta el código desde una dirección especificada.

Este comando pide, por medio de '**:**', un número hexadecimal - una vez entrado el número, la pila interna se reasigna, la pantalla se limpia y se transfiere la ejecución a la dirección especificada. Si desea volver al 'panel frontal' después de ejecutar el código, asigne un '*punto de ruptura*' (ver comando '**W**') en el punto donde desee volver al display.

Ejemplo:

J:B000 <ENTER> ejecuta el código empezando en #8000.

Puede anular este comando antes de terminar la dirección utilizando **CAPS SHIFT 5**.

Observe que '**J**' modifica todos los registros del Z80 antes de ejecutar el código; así el programa en código máquina no hará referencias a los valores almacenados en los registros. Si desea ejecutar código máquina con los registros asignados a unos valores particulares, debe usar el comande **SYMBOL SHIFT K**.

'SYMBOL SHIFT K'

Continúa la ejecución desde la dirección almacenada actualmente en el Contador de programa (*PC*).

Este comando probablemente se utilizará la mayoría de las veces en conjunción con el comando '**W**'. Un ejemplo ayudará a clarificar este tratamiento:

Digamos que utilizamos '*paso a paso*' (utilizando '**Z**') a través del código dado debajo, y ha llegado a la dirección #8920. Ahora no le interesa pasar por la subrutina que está en #9000,pero quiere ver como quedan los '*filas*' (banderines indicadores) después de la llamada a la subrutina en #8800.

```
891E 3EFF           LD    A,-1
8920 CD0090         CALL  #9000
C923 2A0080         LD    HL,(#8000)
8926 7E             LD    A,(HL)
8927 111488         LD    DE,#8814
892A CD0088         CALL  #8800
892D 2003           JR    NZ,lab1
892F 320280         LD    (#8002),A
8932 211488 lab1    LD    HL,#8814
```

Proceda como sigue: active un '*punto de ruptura*', utilizando '**W**' en la dirección #892D (recuerde usar primero '**M**' para asignar el Puntero de Memoria) y luego ejecute un comando '**K**'. La ejecución continúa desde la dirección almacenada en el *PC* la cual, en este caso, es #8920. La ejecución continuará hasta la dirección en la cual ha sido asignado el '*punto de ruptura*' (#892D) en cuyo punto, se actualizará el display y podrá inspeccionar el estado de los banderines, etc. después de la llamada a la subrutina de #8800. Luego puede reanudar el '*paso a paso*' a través del código.

Por tanto, '**K**' es útil para ejecutar el código sin reasignar la pila o sin modificar los registros, como hace '**J**'.

'L'

Tabula, o lista, un bloque de memoria empezando desde la dirección actualmente almacenada en el Puntero de Memoria.

'**L**' limpia la pantalla y muestra la representación hexadecimal y los equivalentes ASCII de los 80 bytes de memoria empezando desde el valor actual. del Puntero de Memoria. Las direcciones se mostrarán bien en hexadecimal o bien en decimal dependiendo el estado actual del '*panel frontal*' (ver **^3** anteriormente). El display consiste en 20 filas con 4 bytes por filas, el ASCII se muestra al final de cada fila. Para el display del ASCII, los valores por encima de 127 son decrementados en 128 y los valores entre 0 y 31 inclusive se muestran como '**.**'.

Al final de una página del listado tiene la opción de volver al '*panel frontal*' principal pulsando **CAPS SHIFT 5** o continuar con la siguiente página de 80 bytes pulsando cualquier otra tecla distinta de **CAPS SHIFT 1**.

'M'

Asigna una dirección especificada al Puntero de Memoria.

Se le pide con ':' que entre una dirección hexadecimal. El Puntero de Memoria se actualiza con la dirección entrada y el display de memoria del '*panel frontal*' cambia de acuerdo a esto.

'**M**' es esencial como preludio de entrar código, tabulando memoria, etc.

'N'

Busca la siguiente localización de la última cadena hex especificada por el comando '**G**'.

'**G**' le permite definir una cadena y luego busca la primera localización de dicha cadena; si quiere buscar posteriores localizaciones de la cadena utilice '**N**'.

'**N**' empieza buscando desde el Puntero de Memoria y actualiza el display de memoria cuando se encuentra la siguiente localización de la cadena.

'0'

Va al destino de un desplazamiento relativo.

El comando toma el byte direccionado por el Puntero de Memoria, lo trata como un desplazamiento relativo y actualiza el display de memoria de acuerdo a lo realizado. Ejemplo:

Digamos que el Puntero de Memoria está asignado a #6800 y que el contenido de las posiciones #67FF y #6800 son #20 y #16 respectivamente - esto podría interpretarse como una instrucción JR NZ,$+24. Para buscar donde iría esta bifurcación en una condición de no-cero, simplemente pulse '**0**' cuando el Puntero de Memoria está direccionando el byte de desplazamiento #16. El display se actualizará entonces al centro de #6817, el destino requerido de la ramificación.

Recuerde que los desplazamientos relativos mayores de #7F (127) se tratan como negativos por el procesador Z80; '**0**' lo tiene en cuenta.

Ver también el comando '**U**' en conexión con '**0**'.

'P'

Llena la memoria entre unos límites especificados con un byte especifico.

'**P**' pregunta '**First:**','**Last:**' y '**With:**'. Entre números hexadecimales en respuesta a estas preguntas; respectivamente, las direcciones de principio y final (inclusive) del bloque que desea llenar y el byte con el que quieres llenar el bloque de memoria. Ejemplo:

P
First: **7000 <ENTER>**
Last: **77FF <ENTER>**
With: **55 <ENTER>**

llenará las posiciones #7000 a #77FF (inclusive) con el byte #55 ('**U**').

Si la dirección inicial es mayor que la final, se anulará '**J**'.

'Q'

Cambia el conjunto de registros.

Al entrar en el '*panel frontal*' el conjunto de registros mostrado es el conjunto Standard3 de registros (AF, HL, DE, BC). El uso de '**Q**' mostrará el conjunto alternativo de registros (AF', HL', DE', BC') el cual se distingue del Standard por la comilla simple "'" después del nombre de registro.

Si se utiliza '**Q**' cuando se muestra el conjunto alternativo, se mostrará el conjunto Standard.

'SYMBOL SHIFT T'

Asigna un '*punto de ruptura*' después de la instrucción en curso continúa la ejecución.

Ejemplo:

```
9000 B7           OR    A
9001 C20098       CALL  NZ,#9800
9004 010000       LD    BC,0

9800 21FFFF       LD    HL,-1
```

Está utilizando '*paso a paso*' en el código anterior y ha alcanzado #9001 con un valor no-cero en el registro *A*, por tanto el banderín *Zero* tendrá un estado *NZ* después de la Instrucción *OR A*. Si ahora utiliza '**Z**' para continuar el '*paso a paso*', la

ejecución continuará en la dirección #9800, la dirección de la subrutina. Si no desea continuar con paso simple a través de esta subrutina, ejecute el comando **^T** cuando esté en la dirección #9001 y el *CALL* será acatado automáticamente y la ejecución se detendrá en la dirección #9004 para que continúe con paso a paso.

Recuerde, **^T** asigna un punto de ruptura después de la instrucción actual y luego ejecuta el comando '**K**'.

Ver el comando '**Z**' para un ejemplo extendido de paso a paso.

'T'

Desensambla un bloque de código, opcionalmente por impresora.

Primero se le pide que entre las direcciones '*First:*' y '*Last:*' del código que desea desensamblar - entre estas en hexadecimal. Si la dirección inicial es mayor que la final, el comando se anula. Después de entrar estas direcciones se le preguntará '*Printer?*'; responda '**Y**' (**CAPS SHIFT + Y**) para dirigir el desensamblado a su impresora, o cualquier otro valor para mandar la salida a la pantalla.

Ahora se le indica '*Text:*' para entrar, en hexadecimal, la dirección inicial de algún fichero de texto que desee que produzca el desensamblador. Si no quiere que se genere ningún fichero, pulse simplemente <ENTER> después de esta indicación. Si especifica una dirección, se producirá un fichero de texto del desensamblado, empezando en esa dirección, en forma utilizable por GENS3. Si quiere utilizar un fichero con GENS3 debe generarlo, o moverlo, en la primera dirección dada por el comando '**X**' del editor del ensamblador, ya que esta es la dirección de inicio del texto esperado por GENS3. Debe también decirle a GENS3 dónde está el final del fichero de texto; hágalo tomando la dirección de '*End of text*' dada por el desensamblador (ver más adelante) y poniéndola en la posición *TEXTEND* de GENS3 - ver manual de GENS3. Luego debe entrar en el GENS3 por la '*entrada en caliente*', para preservar el texto.

Si, en cualquier momento cuando está generando un fichero de texto, el texto sobreescribiera MONS3, se detendría el desensamblado. Pulsa alguna tecla para volver al panel frontal.

Si especificó una dirección de fichero, ahora se le pide que entre una dirección '*Workspace:*' - esto sería el principio de un área de memoria que se utiliza como una tabla de símbolos primitiva para las etiquetas generadas por el desensamblador. La cantidad de memoria requerida es de 2 bytes para cada etiqueta generada. Si pulsa simplemente <ENTER>, se asume una dirección de #6000 (hex).

Después de esto, se le piden repetidamente las direcciones

'*First:*' y '*Last:*' (inclusive) de cualquier área de datos que exista en el bloque que deseas desensamblar. Áreas de datos son áreas de, digamos, texto que no deseas que sean interpretadas como instrucciones del Z80 - en su lugar, estas áreas de datos hacen que se generen pseudo-nemónicos (*DEFB*). Si el valor del byte de datos está entre 32 y 127 (#20 y #7F) inclusive, se dará la interpretación ASCII del byte p.e. #41 se cambia a "*A*" después de un *DEFB*. Cuando ha terminado de especificar áreas de datos, o si no desea especificar ninguna, simplemente escriba <ENTER> en respuesta a ambas preguntas. El comando '**T**' utiliza un área al final de MONS3 para almacenar las direcciones del área de datos y de esta forma puede asignar tantas áreas de datos como memoria disponible hay; cada área de datos requiere 4 bytes de almacenamiento. Observe que utilizando '**T**' destruye cualquier punto de ruptura que estuviese previamente asignado - ver el comando '**W**'.

La pantalla se limpiará ahora. Si pidió que se creara un fichero de texto, habrá un corto retardo (dependiendo de lo grande que sea la sección de memoria que desea desensamblar) mientras se construye la tabla de símbolos. Habiendo hecho esto, el listado desensamblado aparecerá en la pantalla o impresora - puede detener el listado al final de una línea pulsando <ENTER> o <SPACE>, posteriormente pulse **CAPS SHIFT 5** para volver al '*panel frontal*' o cualquier otra tecla (excepto **CAPS SHIFT 1**) para continuar el desensamblado. Si se encuentra un código de operación no válido, es desensamblado como NOP e indicado con un asterisco '*****' después del código en el listado.

Al final del desensamblado se detendrá el display y, si ha pedido que se creara un fichero de texto, se mostrará el mensaje '*End of text xxxxx*'; xxxxx es la dirección en hexadecimal o decimal que deberá POKEarse (primero el byte menos significativo) en la posición *TEXTEND* de GENS3 para que el ensamblador pueda coger este fichero desensamblado mediante una '*entrada en caliente*'. Cuando el desensamblado haya terminado, pulse alguna tecla para volver al panel frontal, aparte de **CAPS SHIFT 1** que le devolverá al BASIC.

Las etiquetas se generan, cuando se requieren (p.e. en C30078), en la forma *Lxxxx*, donde '*xxxx*' es la dirección absoluta en hexadecimal de la etiqueta, pero solo si la dirección concerniente está en los límites del desensamblado. Si la dirección sale fuera de este rango, no se generará ninguna etiqueta, simplemente se dará la dirección hexadecimal o decimal. Por ejemplo, si estábamos desensamblando entre #7000 y #8000, la instrucción *C30078* sería desensamblada como JP L7800; por otra parte, si estábamos desensamblando entre #9000 y #9800, la instrucción sería desensamblada como JP 7800 o JP 30720 si se está utilizando un display decimal. Si se ha hecho referencia a una dirección particular en una instrucción de dentro del desensamblado, su etiqueta aparecerá en el campo de etiquetas (antes del nemónico) del desensamblado de la instrucción en esa dirección pero sólo si el listado va dirigido a un fichero de texto. Ejemplo:

T
First: **8B <ENTER>**
Last: **9E <ENTER>**
Printer? **Y**
Text: **<ENTER>**
First: **95 <ENTER>**
Last: **9E <ENTER>**
First: **<ENTER>**
Last: **<ENTER>**

```
0088  FE16     CP    #16
008D  3801     JR    C,L0090
008F  23       INC   HL
0090  37       SCF
0091  225D5C   LD    (#5C5D),HL
0094  C9       RET
0095  BF524E   DEFB  #BF,"R","N"
0098  04494E   DEFB  #C4,"I","N"
0098  484559   DEFB  "K","E","Y"
009E  A4       DEFB  #A4
```

'U'

Usado en conjunción con el comando '**0**'.

Recuerde que '**0**' actualiza el display de la memoria de acuerdo a un desplazamiento relativo, es decir muestra el efecto de una instrucción *JR* o *DJNZ*. '**U**' se usa para actualizar el display de memoria hacia atrás donde fue ejecutado el último '**0**'. Ejemplo:

```
  7200 47         71F3 77
  7201 20         71F4 C9
>7202 F2<       >71F5 F5<
  7203 06         71F6 C5
display 1       display 2
```

Está en *display 1* y desea saber donde va el salto relativo 20F2. Así, pulse '**0**' y el display de memoria se actualiza al *display 2*. Ahora investiga el código siguiente a #71F5 por un momento y luego desea volver al código siguiente al salto relativo original para ver que pasa si el banderín de cero está activado. Por tanto, pulse '**U**' y el display de memoria volverá a *display 1*.

Observe que solo puede usar '**U**' para volver a la última aparición del comando '**0**', todas las utilizaciones anteriores de '**0**' se pierden.

'V'

Utilizado en conjunción con el comando '**X**'.

'**V**' es similar al comando '**U**' en su efecto, excepto que actualiza el display de memoria a donde estaba antes de que el último comando '**X**' fuera ejecutado. Ejemplo:

```
 8702 AF          842D 18
 8703 CD          842E A2
>8704 2F<        >842F E5<
 8705 44          8430 21
display 1        display 2
```

Está en el *display 1* y desea echar un vistazo a la subrutina que está en #842F. Por tanto pulse '**X**' con el display centrado como se muestra; el display de memoria entonces pasa al *display 2*. Mira la subrutina por un momento y luego desea volver al código situado después de la llamada original a la subrutina. Por tanto pulse '**V**' y reaparecerá el *display 1*.

Como con '**U**' puede utilizar este comando solo para alcanzar la dirección en la cual el último comando '**X**' fue ejecutado, todas las direcciones anteriores en las que se utilizó '**X**' se pierden.

'**W**'

Activa un punto de ruptura en el Puntero de Memoria. Un *'punto de ruptura*', tal como se entiende en MONS3, es simplemente una instrucción *CALL* a una rutina de MONS3 que muestra el '*panel frontal*' permitiendo así al programador detener la ejecución de un programa e inspeccionar los registros del Z80, banderines y muchas posiciones de memoria. Así, si desea detener la ejecución de un programa en #9876, digamos, utilice el comando '**M**' para activar el Puntero de Memoria a #9876 y luego utilice '**W**' para activar un punto de ruptura en esa dirección. Los 3 bytes de código que estaban originalmente en #9876 se salvan y son reemplazados por una instrucción *CALL* que detiene la ejecución cuando se alcanza. Cuando esta instrucción *CALL* es alcanzada causa que los 3 bytes originales sean reemplazados en #9876 y el '*panel frontal*' se muestra con todos los registros y banderines en el estado en que estaban justo antes de que se ejecutara el punto de ruptura. Ahora puede utilizar alguna de las facilidades de MONS3 de la forma usual.

Notas:
MONS3 utiliza el área, al final de sí mismo, que originalmente contiene las direcciones de reasignación para almacenar la información de los puntos de ruptura. Esto significa que puede activar tantos puntos de ruptura como memoria disponible haya; cada punto de ruptura requiere 5 bytes de almacenamiento. Cuando ejecuta un punto de ruptura, MONS3 restaurará automáticamente el contenido de memoria que existía antes de la asignación de ese punto de ruptura. Observe que, ya que el comando '**T**' también utiliza este área, todos los puntos de ruptura se pierden cuando se utiliza el comando '**T**'. Los puntos de ruptura sólo pueden asignarse en RAM. Ya que un punto de ruptura consiste de una instrucción *CALL* de 3 bytes se ha de tener cierto cuidado en ciertos casos excepcionales, p.e. considere el código:

```
 8000 3E          8008 00
 8001 01          8009 00
 8002 18          800A 06
 8003 06          800B 02
>8004 AF<         800C 18
 8005 0E          800D F7
 8006 FF          800E 06
 8007 01          800F 44
```

Si activa un punto de ruptura en #8004 y empieza la ejecución del código desde la posición #8000, el registro *A* se cargará con el valor 1, la ejecución se transfiere a #800A, el registro *B* se carga con el valor 2 y la ejecución se transfiere a la posición #8005. Pero #8005 ha sido sobreescrita con el byte menos significativo del punto de ruptura y de esta forma habremos modificado el código y se producirán resultados impredecibles. Este tipo de situación es inusual pero debe intentar evitarla - en este caso el '*paso a paso*' del código proporcionará la respuesta; ver el comando '**^Z**' más adelante para un ejemplo detallado del '*paso a paso*'.

'X'

Utilizado para actualizar el Puntero de Memoria con el destino de una instrucción absoluta CALL o JP.

'**X**' toma la dirección de 16 bits especificada por el byte del Puntero de Memoria y el byte del *Puntero de Memoria + 1* y luego actualiza el display de memoria para que esté centrado alrededor de esa dirección. Recuerde que la mitad de orden inferior de la dirección es especificada por el primer byte y la mitad de la dirección de orden superior es dada por el segundo byte (formato Intel). Ejemplo:
digamos que quiere echar un vistazo a la rutina a la que llama el código CD0563; asigne el Puntero de memoria (usando '**M**') para que direccione al 05 de la instrucción *CALL* y luego pulse '**X**'. El display de memoria será actualizado de forma que está centrado en la posición #6305.

Vea también el comando 'V' en conexión con 'X'.

'Y'

Entra ASCII desde el Puntero de memoria.

'**Y**' le da una nueva línea sobre la que puede entrar caracteres *ASCII* directamente del teclado. Estos caracteres introducidos y sus equivalentes decimales son entrados en memoria empezando desde el valor actual del Puntero de memoria. La cadena de caracteres deberá ir terminada por **CAPS SHIFT 5** y **DELETE** (**CAPS SHIFT 0**) puede utilizarse para borrar caracteres de la cadena. Cuando haya acabado de entrar los caracteres ASCII (y haya

pulsado **CAPS SHIFT 5**) el display se actualiza de forma que el puntero de memoria se posiciona después del final de la cadena que se entró en memoria.

'**SYMBOL SHIFT Z**'

Paso a paso.

Antes de usar '**^Z**' (o '**^T**'), el Contador de Programa y el Puntero de memoria, deben asignarse a la dirección de la instrucción que desea ejecutar.

'**Z**' simplemente ejecuta la instrucción actual y luego actualiza el '*panel frontal*' para reflejar los cambios causados por la instrucción ejecutada.

Observe que puede hacer paso a paso en cualquier parte de la memoria (RAM o ROM) pero debe asegurar que las interrupciones no están habilitadas en ningún momento. Ahora siga un ejemplo extendido que clarificará la utilización de muchos de los comandos de depuración disponibles en MONS3. Le instamos a estudiarlo cuidadosamente y a intentar sacarlo por Ud. mismo.

Asumamos que tenemos las 3 secciones de código mostradas abajo en la máquina, la primera sección es el programa principal que siga *HL* y *DE* con números y luego llama a una rutina para multiplicarlos (la segunda sección) con el resultado en *HL* y finalmente llama a una rutina dos veces para sacar el resultado de la multiplicación por la pantalla (tercera sección).

```
7080 2A0072              LD    HL,(#7200) ;SECCION 1
7083 ED5B0272            LD    DE,(#7202)
70S7 CD0071              CALL  Mult
708A 7C                  LD    A,H
708B CD1D71              CALL  Aout
708E 7D                  LD    A,L
708F CD1D71              CALL  Aout
7092 210000              LD    HL,0
.
.
.
7100 AF          Mult    XOR   A ;SECCION 2
7101 ED52                SBC   HL,DE
7103 19                  ADD   HL,DE
7104 3001                JR    NC,Mu1
7106 EB                  EX    DE,HL
7107 B2          Mu1     OR    D
7108 37                  SCF
7109 C0                  RET   NZ
7110 B3                  OR    E
710B 5A                  LD    E,D
710C 2007                JR    NZ,Mu4
```

```
710E EB                     EX    DE, HL
710F C9                     RET
7110 EB          Mu2        EX    DE,HL
7111 19                     ADD   HL, DE
7112 EB                     EX    DE, HL
7113 29          Mu3        ADD   HL, HL
7114 D8                     RET   C
7115 1F          Mu4        RRA
7116 30FB                   JR    NC,Mu3
7118 B7                     OR    A
7119 20F5                   JR    NZ,Mu2
711B 19                     ADD   HL,DE
711C C9                     RET

711D F5          Aout       PUSH  AF ;SECCION 3
711E 0F                     RRCA
711F 0F                     RRCA
7120 0F                     RRCA
7121 0F                     RRCA
7122 CD2671                 CALL Nibble
7125 F1                     POP   AF
7126 E60F        Nibble     AND   %1111
7128 C690                   ADD   A,#90
712A 27                     DAA
712B CE40                   ADC   A,#40
712D 27                     DAA
712E FD213A5C               LD    IY, #5C3A
7132 D7                     RST   #10
7133 C9                     RET
.
.
7200 1B2A                   DEFW  10779
7202 0300                   DEFW  3
```

Ahora deseamos investigar el código anterior ya sea para ver si funciona o bien para ver como lo hace. Podemos hacer esto con el siguiente conjunto de comandos - debe notarse que esto es simplemente una forma de ir a través del código, no es necesariamente eficiente pero servirá para demostrar el '*paso a paso*':

M:7080 <ENTER>	asigna el Puntero de memoria a #7080.
7080.	asigna el Contador de programa a #7080.
^Z	paso a paso.
^Z	paso a paso.
^Z	sigue el CALL.
M:7115 <ENTER>	salta el pre-procesamiento de los números.
W	asigna un punto de ruptura.
^K	continúa la ejecución desde #7100 hasta el punto de ruptura.
^Z	paso a paso.
^Z	sigue el salto relativo.
^Z	paso a paso.
^Z	" "
^Z	" "
^Z	" "
^Z	" "

^Z	paso a paso.
^Z	" "
^Z	vuelve de la rutina de multiplicación.
^Z	paso a paso.
^Z	sigue el CALL.
M:7128 <ENTER>	asigna el Puntero de memoria al byte que interesa.
W	activa un punto de ruptura.
^K	continúa la ejecución desde #711D hasta el punto de ruptura.
^Z	paso a paso.
^Z	" "
^Z	" "
^Z	" "
,	mira la dirección de retorno
W	activa un punto de ruptura ahí
^K	y continúa.
^Z	paso a paso.
,	vuelve de la rutina Aout
W	
^K	
^Z	paso a paso.
^T	acata la llamada completa a Aout.

Por favor, trabaje con el ejemplo anterior, primero escribiendo el código de las rutinas (ver "Modificando la memoria" más adelante) o utilizando GENS3, y luego utilizando los comandos detallados anteriormente. Encontrará el ejemplo invalorable como ayuda para entender como trazar el camino a través de un programa.

SYMBOL SHIFT P

Este comando es exactamente el mismo que el comando '**L**'ist excepto que la salida va a la impresora en vez de ir a la pantalla. Recuerde que, al final de una página, pulse **CAPS SHIFT 5** para volver al '*panel frontal*' o cualquier otra tecla (excepto **CAPS SHIFT 1**) para obtener otra página.

Modificando la memoria.

El contenido de las direcciones dadas por el Puntero de memoria puede modificarse entrando un número hexadecimal seguido por un terminador (ver Sección 1). Los dos dígitos hexadecimales menos significativos (si solo se entra un dígito se rellena a la izquierda con un cero) se entran en la posición direccionada por el Puntero de Memoria y entonces se acata el comando (si existe) especificado por el terminador. Si el terminador no es un comando válido, se ignora.

Ejemplos:

F2 ENTER	Se entra #F2 y el Puntero de Memoria avanza en 1.
123 CAPS SHIFT 8	Se entra #23 y el Puntero de Memoria avanza en 8.
EM:E00_	Se entra #0E en el Puntero de Memoria actual y luego éste se actualiza a #E00. Observe que se ha utilizado un espacio para terminar el comando '**M**'.
8C0	Se entra #8 y el Puntero de Memoria se actualiza (debido al comando '**0**') al destino del equivalente relativo #8C, es decir a su valor actual - 115.
2A5D_	Se entra #5D y el Puntero de Memoria no cambia, ya que el terminador es un espacio, no un comando.

Modificando Registros.

Si se entra un número hexadecimal en respuesta al indicador '**>**' y se termina por un punto, "**.**", el número especificado se entrará en el registro del Z80 direccionado por la flecha '**→**'.

Al entrar en MONS3, '**→**' apunta al Contador de Programa (PC) y así, utilizando '**.**' como terminador de un número hexadecimal inicialmente modificará el Contador de Programa. Si desea modificar cualquier otro registro, utilice '.' sólo (no como terminador) y el puntero '**→**' recorrerá los registros PC hasta *AF*. Observe que no es posible direccionar (y por tanto cambiar) el Puntero de Pila (SP) o los registros *IR*.

Ejemplos:

Asuma que el puntero de registro '**→**' está inicialmente direccionando el PC.

.	apunta a IY
.	apunta a IX.
0.	asigna cero a 1X.
.	apunta e HL.
123.	asigna # 123 a HL.
.	apunta a DE.
.	apunta a 9C.
E2A7.	asigna #E2A7 a 8O.
.	apunta a AF.
FF00.	asigna #FP a A y desactiva todos los banderines.
.	apunta a PC.
8000.	asigna #8000 a PC.

Observe que '**.**' también puede utilizarse para modificar el conjunto alternativo de registros si este está en el display. Utilice el comando '**Q**' para cambiar el display del conjunto de registros.

APÉNDICE - UN EJEMPLO DEL DISPLAY DEL PANEL FRONTAL.

```
                710C  2007      JR   NZ, 7115
                 PC 710C   20 07 EB C9 EB 19 EB
                 SP DOAF   8A 70 06 03 0A 03 0D
                 IY CF6A   0D 11 0C 0F 09 18 18
                 IX D09F   04 03 04 00 00 00 1B
                 HL 2A1B   DF FE 29 28 02 CF 02
                 DE 0000   F3 AF 11 FF FF C3 CB
                 BC 0004   FF C3 CB 11 2A 5D 5C
                 AF 0304       V
                 IR 3F7C

                7100  AF   7108  37   7110  EB
                7101  ED   7109  C0   7111  19
                7102  52   710A  B3   7112  EB
                7103  19   710B  5A   7113  29
                7104  30  >710C  20<  7114  D8
                7105  01   710D  07   7115  1F
                7106  EB   710E  EB   7116  30
                7107  B2   710F  C9   7117  FB
              >
```

Lo mostrado arriba es un típico display del '*panel frontal*' - el display es uno obtenido mientras utilizamos '*paso a paso*' con la rutina Mult dada en el ejemplo del comando '**SYMBOL SHIFT Z**'.

Las primeras 9 líneas del display contienen los registros del Z80; el nombre del registro primero (*PC* a *IR*), luego (para *PC* a *BC*) el valor almacenado en el registro, y finalmente el contenido de las siete posiciones de memoria empezando desde la dirección almacenada en el registro. El registro *Flag* se decodifica para mostrar los banderines que están activados en el orden de bit en que se usan en el registro - si el registro *Flag* estaba activado a #FF, el display que sigue a *AF*, se verá como 00FF SZ HVNC, indicando que los banderines de signo, cero, medio acarreo, paridad/overflow, suma/resta y acarreo están activados.

Un Puntero de registros '**→**' apunta al registro actualmente direccionado.

El display de 24 bytes de memoria siguiente al display de registros está organizado como la dirección (2 bytes, 4 caracteres) seguida del contenido (1 byte, 2 caracteres) de la memoria en esa dirección. El display es centrado alrededor del valor del Puntero de memoria, indicado por '**> <**'.

Los comandos (ver Sección 2) se entran en la línea inferior de la pantalla en respuesta al indicador '**>**'. El display se actualiza después de procesar cada comando.

USO DEL MICRODRIVE

Realizando Copia de Seguridad

Cargue el GENS3M2 o el MONS3M2 tal como se indica al principio del manual, pero dejando más memoria libre por debajo del *CLEAR* y sin ejecutar el programa. Una vez cargado podrá hacer una Copia de Seguridad en cassette o Microdrive del siguiente modo:

GENS3M2

SAVE "GENS3M2" CODE XXXXX,10034	para cassette
SAVE *"m";1;"GENS3M2" CODE XXXXX,10034	para Microdrive

MONS3M2

SAVE "MONS3M2" CODE XXXXX,6068	para cassette
SAVE *"m";1;"MONS3M2" CODE XXXXX,6068	para Microdrive

donde *XXXXX* es la dirección donde ha cargado el programa.

Nuevos comandos

No hay comandos añadidos para Microdrive en MONS3M2, el código ha sido simplemente alterado para que sea posible hacer paso a paso del código que utiliza la ROM del Interface 1. Por favor, observe que no es posible hacer paso a paso de la ROM del Interface 1 por si mismo.

Un aspecto que se ha añadido es relativo a la acción a tomar cuando se encuentra un punto de ruptura. Anteriormente, cuando se encontraba un punto de ruptura, se producía una interrupción en el display del panel frontal; en la versión 3.2M, cuando se encuentra un punto de ruptura, se hace sonar un tono corto y la ejecución se detiene, esperando a que pulse una tecla. Cuando pulsa una tecla se entra el panel frontal.

Además, ahora no es necesario tener el mismo valor en el Puntero de memoria y el Contador de programa cuando utilizamos paso a paso.

Las opciones extra disponibles en GENS3M2 (comparado con GENS3) son las que siguen:

Los comandos 'P' y 'G' del editor.

Estos comandos son idénticos para '**P**'oner o '**G**'(cargar) texto a y desde el cassette, pero han sido modificados de forma que, si los dos primeros caracteres del nombre del fichero son un dígito (1-8 inclusive) seguido por dos puntos ':', el texto será salvado o cargado desde el Microdrive oportuno. Así, para almacenar las líneas 10 a 190 de un fichero de texto al

Microdrive 1, utilice simplemente: **P1,P190,1:TEST**
Y el fichero se almacenará en el Microdrive 1 con el nombre 'TEST'.

Observe que, cuando '**P**'onemos (Save), si el fichero ya existe, se le preguntará '*File Exists Delete (Y/N)?*' – responda **Y** para borrar el fichero y continuar o cualquier otra tecla para volver al editor sin borrar el fichero. Si, cuando '**G**'-cargamos (*Load*) información, el fichero no existe se mostrará el mensaje '*Absent*'. Debe especificar un nombre de fichero si desea cargar un fichero desde un cartucho de Microdrive – no puede cargar el primer fichero ya que ninguno lo es.

Si no tiene Microdrives conectados a su sistema, debe asegurarse de que el segundo carácter de cualquier nombre de fichero que utilice no sean los dos puntos ':'.

El comando '*F' del Ensamblador.

'***F**' se utiliza para 'incluir' texto desde un elemento de almacenamiento mientras ensamblamos. En GENS3M2 este comando se ha extendido para permitirle incluir texto desde un cartucho de Microdrive – de nuevo el método es incluir un número de Microdrive (1 – 8 inclusive) antes del nombre de fichero. En cassette debe utilizar el comando '**T**' del editor para salvar el texto que desea incluir posteriormente, y el uso de '***F**' automáticamente desactiva cualquier otro comando del ensamblador. En Microdrive estas dos restricciones pueden ser aminoradas. Puede incluir un fichero de texto que fue almacenado en cartucho utilizando el comando 'P' y todos los comandos están disponibles mientras incluimos. Así, para incluir el fichero de texto '*TEST*' (ver el ejemplo anterior) puede ser algo como esto:

```
          LD    A,B
          CALL  AOUT
*H  Incluye el fichero 'TEST' desde Microdrive
          *F 1:TEST
*H  Siguiente rutina
```

Debe especificar un nombre de fichero cuando utilizamos '***F**' con Microdrives y, si el código incluido contiene un error, debe continuar con el ensamblado. No intente volver al editor (con '**E**') mientras incluimos texto desde un Microdrive.

El comando 'O' del editor.

El comando '**O**' ha sido añadido para permitirle salvar código objeto directamente a cassette o Microdrive. El formato de estecomando es **O,,nombre de fichero** donde el nombre del fichero puede tener hasta 8 caracteres. Si el nombre de fichero empieza con un número seguido por dos puntos '**:**', el código objeto será salvado en el Microdrive especificado por el número que precede a los dos puntos; en caso contrario el código se salvará en cinta.

Notará que solo puede grabarse el último bloque de código producido por GENS3M2 de esta forma, sólo se salva el código producido después del último *ORG*.

El código deberá haberse producido en la memoria del SPECTRUM antes de que sea salvado utilizando '**O**'.

El comando 'V' del editor.

Este comando se ha modificado para mostrar el delimitador actual (inicialmente una coma) así como los números de línea y cadenas por defecto. Es útil para saber cual es el delimitador si lo ha cambiado inadvertidamente utilizando el comando '**S**'.

CATálogo extendido.

Presentamos un listado ensamblador de un programa que produce un CATálogo de un cartucho de Microdrive similar al producido por el CAT del BASIC, pero con información extra como:

Tipo de fichero:
D – para un fichero de tipo datos o print
P – para un fichero de programa
B – para un fichero CODE
S – para una matriz alfanumérica
N – para una matriz numérica

También para un fichero CODE, se muestran su longitud y su dirección de inicio en decimal, mientras que para un fichero de programa, se muestra, en decimal, su longitud y su dirección de auto-ejecución.

Para utilizar el programa escriba los códigos hexadecimales directamente (utilizando MONS3M2 o POKEándolos desde BASIC) o utilice GENS3M2 para ensamblar el programa fuente. Note que el programa está actualmente colocado en la dirección 60000. Por supuesto puede cambiar esto si reensambla.

Una vez el código está en el ordenador puede ejecutarlo desde BASIC o hacerlo desde el editor del GENS3M2.

Para utilizarlo desde BASIC, primero POKEe el número de Microdrive que desea catalogar en la posición 60345 (asumiendo que el código empieza en 60000) y luego emplee **RANDOMIZE USR 60000**.

Para usar el código desde GENS3M2 debe usar el comando '**Z**' para saltar al código del CATálogo. Para hacer esto, cargue GENS3M2 y luego *POKEe la dirección del catálogo extendido + 2* en las posiciones *Principio de GENS3M2 + 7790* y *Principio de GENS3M2 + 7791* (byte de orden inferior primero). Por ejemplo, digamos que ha cargado GENS3M2 en 26000 y la rutina del catálogo extendido está en 60000. Por tanto para efectuar el '*parche*' simplemente haga **POKE 33790, 98 , POKE 33791, 234** y luego entre GENS3M2 en la forma normal.

Ahora puede usar **Zn** desde el editor para CATalogar el drive número *n*.

Le ofrecemos el listado del programa del CATálogo extendido:

Catálogo extendido de Microdrive Hisoft GEN Assembler

```
       1 *H Catalogo extendido de Microdrive
       2
       3 ;Realizado con el ensamblador GEN
       4
       5 ;Copyright Hisoft 1984
       6 ;Muchas gracias a Andrew Pennell por su "biblia" del
       7 ;Microdrive y la rutina Stream 14.
       8
       9 ;Etiquetas
      10
000D  11 CR     EQU  13
      12
0002  13 PRINT  EQU  2
0043  14 RECFLG EQU  67
      15
15D4  16 WAIT_K EQU  #15D4
1655  17 MAKE_S EQU  #1655
      18
5C16  19 STRMS  EQU  #5C16
5C3A  20 ERR_NR EQU  #5C3A
5C4F  21 CHANS  EQU  #5CF4
5C53  22 PROG   EQU  #5C53
      23
5CD6  24 D_STR1 EQU  #5CD6
5CD8  25 S_STR1 EQU  #5CD8
5CDA  26 N_STR1 EQU  #5CDA
5CDC  27 T_STR1 EQU  #5CDC
5CE6  28 HD_00  EQU  #5CE6
5CE7  29 HD_0B  EQU  #5CE7
5CE9  30 HD_0D  EQU  #5CE9
5CEB  31 HD_0F  EQU  #5CEB
5CED  32 HD_11  EQU  #5CED
      33
```

```
0022           34 OPEN_M EQU  #22
0023           35 CLOSE_ EQU  #23
0031           36 NEWVAR EQU  #31
0032           37 SHADOW EQU  #32
               38
               39 ;Solo para GENS3M2
               40
1A0C           41 NUM1of EQU  7297
               42
               43
EA60           44        ORG  60000
               45
               46 ;Punto de entrada BASIC
               47
EA60 1817      48        JR   BasEnt
               49
               50 ;Punto de entrada GENS3M2
               51
EA62 E1        52        POP  HL
EA63 54        53        LD   D,H
EA64 5D        54        LD   E,L
EA65 E5        55        PUSH HL
EA66 21811C    56        LD   HL,NUM1of
EA69 19        57        ADD  HL,DE
EA6A 7E        58        LD   A,(HL)
EA6B 23        59        INC  HL
EA6C B6        60        OR   (HL)
EA6D 2002      61        JR   NZ,NumSet
EA6F 3E01      62        LD   A,1
EA71 E60F      63 NumSet AND  %00001111
EA73 2600      64        LD   H,0
EA75 6F        65        LD   L,A
EA76 22B9EB    66        LD   (DRIVE),HL
               67
               68 ;Inicializa
               69
EA79 210AEC    70 BasEnt LD   HL,SPACE
EA7C 2208EC    71        LD   (POINTER),HL
EA7F FD213A5C  72        LD   IY,#5C3A
               73
EA83 21C5EB    74        LD   HL,SIGNON
EA86 0634      75        LD   B,SIGNEND-SIGNON
EA88 CD9AEB    76        CALL WRstring
               77
               78 ;Prepara nuevo canal y lo asigna al Stream 14
               79
EA8B 2A535C    80        LD   HL,(PROG)
EA8E 2B        81        DEC  HL
EA8F E5        82        PUSH HL
EA90 010B00    83        LD   BC,11
EA93 CD5516    84        CALL MAKE_SPACE
EA96 2191EB    85        LD   HL,CH14out
```

```
EA99 D1         86        POP  DE
EA9A D5         87        PUSH DE
EA9B EB         88        EX   DE,HL
EA9C 73         89        LD   (HL),E
EA9D 23         90        INC  HL
EA9E 72         91        LD   (HL),D
EA9F 23         92        INC  HL
EAA0 EB         93        EX   DE,HL
EAA1 21FDEB     94        LD   HL,C14INFO
EAA4 010900     95        LD   BC,11-2
EAA7 EDB0       96        LDIR
EAA9 E1         97        POP  HL
EAAA 23         98        INC  HL
EAAB ED4B4F5C   99        LD   BC,(CHANS)
EAAF B7        100        OR   A
EAB0 ED42      101        SBC  HL,BC
EAB2 22325C    102        LD   (STRMS+28),HL
               103
               104 ;Ahora lee el catalogo simple
               105
EAB5 D9        106        EXX
EAB6 E5        107        PUSH HL
EAB7 D9        108        EXX
               109
EAB8 CF        110        RST  8
EAB9 31        111        DEFB NEWVARS
               112
EABA 3E0E      113        LD   A,14
EABC 32D85C    114        LD   (S_STR1),A
EABF 2AB9EB    115        LD   HL,(DRIVE)
EAC2 22D65C    116        LD   (D_STR1),HL
EAC5 2A06EC    117        LD   HL,(CAT)
EAC8 22ED5C    118        LD   (HD_11),HL
EACB FB        119        EI
EACC CF        120        RST  8
EACD 32        121        DEFB SHADOW
               122
               123 ;Ahora procesa el catalogo simple
               124
EACE 210AEC    125        LD   HL,SPACE
EAD1 060B      126        LD   B,11
EAD3 CD9AEB    127        CALL WRstring
EAD6 060F      128 CatL1  LD   B,15
EAD8 C5        129 CatLoo PUSH BC
EAD9 23        130        INC  HL
EADA 7E        131        LD   A,(HL)
EADB FE0D      132        CP   CR         ;final?
EADD 2879      133        JR   Z,CatEnd
EADF 2B        134        DEC  HL
EAE0 060B      135        LD   B,11
EAE2 E5        136        PUSH HL
EAE3 CD9AEB    137        CALL Wrstring
EAE6 E3        138        EX   (SP),HL
EAE7 23        139        INC  HL
EAE8 EB        140        EX   DE,HL
EAE9 21DC5C    141        LD   HL,T_STR1
```

```
EAEC 73       142        LD   (HL),E
EAED 23       143        INC  HL
EAEE 72       144        LD   (HL),D
EAEF 2AB9EB   145        LD   HL,(DRIVE)
EAF2 22D65C   146        LD   (D_STR1),HL
EAF5 210A00   147        LD   HL,10
EAF8 22DA5C   148        LD   (N_STR1),HL
EAFB FB       149        EI
EAFC CF       150        RST  8
EAFD 22       151        DEFB OPEN_M
EAFE CDB5EB   152        CALL Space
EB01 DDCB4356 153        BIT  PRINT,(IX+RECFLG)
EB05 2007     154        JR   NZ,NotPrint
EB07 3E44     155        LD   A,"D"
EB09 CDACEB   156        CALL CONOUT
EB0C 1841     157        JR   CatBack
EB0E DDE5     158 NotPri PUSH IX
EB10 D1       159        POP  DE
EB11 215200   160        LD   HL,82
EB14 19       161        ADD  HL,DE
EB15 EB       162        EX   DE,HL
EB16 1A       163        LD   A,(DE)
EB17 13       164        INC  DE
EB18 21F9EB   165        LD   HL,TYPETAB
EB1B 4F       166        LD   C,A
EB1C 0600     167        LD   B,0
EB1E 09       168        ADD  HL,BC
EB1F 7E       169        LD   A,(HL)
EB20 CDACEB   170        CALL CONOUT
EB23 CDB5EB   171        CALL Space
EB26 EB       172        EX   DE,HL
EB27 79       173        LD   A,C
EB28 B7       174        OR   A
EB29 2013     175        JR   NZ,NotProg
EB2B 23       176        INC  HL
EB2C 23       177        INC  HL
EB2D 23       178        INC  HL
EB2E 23       179        INC  HL
EB2F 5E       180        LD   E,(HL)
EB30 23       181        INC  HL
EB31 56       182        LD   D,(HL)
EB32 23       183        INC  HL
EB33 CD66EB   184        CALL DEOUTS
EB36 5E       185        LD   E,(HL)
EB37 23       186        INC  HL
EB38 56       187        LD   D,(HL)
EB39 CD66EB   188        CALL DEOUTS
EB3C 1811     189        JR   CatBack
EB3E FE03     190 NotPro CP   3
EB40 200D     191        JR   NZ,CatBack
EB42 5E       192        LD   E,(HL)
EB43 23       193        INC  HL
EB44 56       194        LD   D,(HL)
EB45 23       195        INC  HL
EB46 CD66EB   196        CALL DEOUTS
EB49 5E       197        LD   E,(HL)
```

```
EB4A 23       198        INC  HL
EB4B 56       199        LD   D,(HL)
EB4C CD66EB   200        CALL DEOUTS
EB4F CF       201 CatBac RST  8
EB50 23       202        DEFB CLOSE_M
EB51 E1       203        POP  HL
EB52 C1       204        POP  BC
EB53 1083     205        DJNZ CatLoop
EB55 C3D6EA   206        JP   CatL1
              207
```

...
...
...

Por un error de la imprenta (supongo)
aquí falta una página entera con 60 líneas...
(Casi na')

...
...
...

```
EBAA E1       267        POP  HL
EBAB C9       268        RET
              269
              270 ;Salida por stream 2
              271
EBAC F5       272 CONOUT PUSH AF
EBAD 3E02     273        LD   A,2
EBAF CDA2EB   274        CALL ChOpen
EBB2 F1       275        POP  AF
EBB3 D7       276        RST  #10
EBB4 C9       277        RET
              278
              279 ;Saca un espacio por el stream 2
              280
EBB5 3E20     281 Space  LD   A," "
EBB7 18F3     282        JR   CONOUT
              283
              284
              285 ;Almacenamiento
              286
EBB9 0100     287 DRIVE  DEFW 1
              288
EBBB 1027E803 289 TENTAB DEFW 10000,1000,100,10,1
     64000A00
     0100
              290
EBC5 0D       291 SIGNON DEFB CR
EBC6 4869736F 292        DEFM "Hisoft Extended CAT Listing"
     66742045
     7874656E
     64656420
     43415420
     4C697374
     696E67
```

```
EBE1 0D       293        DEFB CR
EBE2 436F7079 294        DEFM "Copyright Hisoft 1984"
     72696768
     74204869
     736F6674
     20313938
     34
EBF7 0D0D     295        DEFB CR,CR
EBF9          296 SIGNEN EQU  $
              297
EBF9 504E5342 298 TYPETA DEFM "PNSB"
              299
EBFD C415     300 C14INF DEFW #15C4
EBFF 5A       301        DEFB "Z"
EC00 28002800 302        DEFW #28,#28,11
     0B00
              303
              304 ;********* Este valor puede cambiar con la ROM
              305 ;          nueva del Interface 1 *************
EC06 581C     306 CAT    DEFW #1C58
              307
              308 ;******************************************
              309
              310
EC08          311 POINTE DEFS 2
EC0A          312 SPACE  DEFS 512
              313

Pass 2 errors: 00
```

IMPORTANTE

CONSULTAS Y MODIFICACIONES

Como es evidente, el precio de este programa no incluye ninguna clase de consulta técnica ni modificaciones en el programa a nivel particular, y por ello hemos elaborado este manual de instrucciones. No obstante, dado que queremos ofrecer el máximo soporte a los usuarios de nuestros programas, contestaremos personalmente o por carta las consultas y ha-remos las modificaciones que el usuario requiera, siempre que estén dentro de unos límites razonables y claros, facturando este servicio según el tiempo y categoría del personal necesario para ello. Las consultas personales sólo se atenderán siempre que se concierten previamente.
Para tener derecho a cualquier consulta, le rogamos que remita inmediatamente el adjunto CUPON DE USUARIO.

CUPON DE USUARIO

Enviar a: VENTAMATIC, c/ Córcega, 89. 08029 - BARCELONA

PROGRAMA .....

ADQUIRIDO EL DIA ..... MES .....

EN EL ESTABLECIMIENTO .....

POR D. .....

CALLE .....

.....

POBLACIÓN ..... PROVINCIA .....